Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
Кафедра систем искусственного интеллекта

БАКАЛАВРСКАЯ РАБОТА

09.03.02.04 «Информационные системы и технологии в медиаиндустрии»

Децентрализованное приложение назначения встреч на мероприятиях

Руководитель _____ К. В. Раевич
подпись, дата

Выпускник _____ С. Е. Мерцев
подпись, дата

Нормоконтролер _____ К. В. Раевич
подпись, дата

Красноярск 2018

Продолжение титульного листа бакалаврской работы по теме «Децентрализованное приложение назначения встреч на мероприятиях»

Консультанты по
разделам:

Нормоконтролер _____ К. В. Раевич

подпись, дата

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт космических и информационных технологий
Кафедра систем искусственного интеллекта

УТВЕРЖДАЮ
Заведующий кафедрой
____ Г. М. Цибульский
подпись
«__» _____2018 г.

ЗАДАНИЕ НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
в форме бакалаврской работы

Студенту Мерцеву Сергею Егоровичу

Группа КИ14-11Б, направление 09.03.02 «Информационные системы и технологии», профиль 09.03.02.04 «Информационные системы и технологии в медиаиндустрии».

Тема выпускной квалификационной работы «Разработка децентрализованного приложения назначения встреч на мероприятиях»

Утверждена приказом по университету № 4533/с от 28.03.2018

Руководитель ВКР Раевич К.В. доцент кафедры систем искусственного интеллекта ИКИТ СФУ.

Исходные данные для ВКР: методические указания руководителя

Перечень разделов ВКР: введение; глава 1. Теоретическая часть; выводы по главе 1; глава 2. Программная часть; выводы по главе 2; заключение; список использованных источников; приложение (плакаты презентации).

Руководитель ВКР	_____	К. В. Раевич
	подпись	
Задание принял к исполнению	_____	С. Е. Мерцев
	подпись	

«12» марта 2018 г.

График

выполнения выпускной квалификационной работы студентом направления 09.03.02 «Информационные системы и технологии», профиля 09.03.02.04 «Информационные системы и технологии в медиаиндустрии» приведен в таблице 1.

Таблица 1 – График выполнения этапов ВКР

Наименование этапа	Срок выполнения этапа	Результат выполнения этапа	Примечание руководителя (отметка о выполнении этапа)
Ознакомление с целью и задачами	12.03 – 16.03	Краткий обзор по теме	Выполнено
Ознакомление со средой разработки	17.03 – 20.03	Установлено необходимое для разработки программное обеспечение	Выполнено
Разработка смарт-контрактов	17.03 – 27.03	Смарт-контракты на языке solidity, выполняющие логику приложения	Выполнено
Разработка веб-интерфейса	28.03 – 24.04	Веб-интерфейс для взаимодействия с пользователем	Выполнено
Совмещение логики веб-интерфейса и смарт-контрактов	25.04 – 25.04	Взаимодействие смарт-контрактов с веб-интерфейсом	Выполнено

Тестирование программного кода на наличие ошибок	25.04 – 25.04	Исправление ошибок в программном коде	Выполнено
Нормоконтроль (Н/К)	6.06 – 19.06	Пояснительная записка, презентация к ВКР	Выполнено
Защита ВКР	21.06	Доклад и презентация по результатам бакалаврской работы	Выполнено

Студент _____ С. Е. Мерцев

подпись

Руководитель ВКР _____ К. В. Раевич

Доцент кафедры СИИ подпись

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время формируется новая модель создания масштабируемых и эффективных приложений, основой для которой является система Bitcoin и технология блокчейн.

Самой распространённой моделью работы различных приложений и интернет-сервисов на данный момент является централизованная. В таких системах вся информация и процессы проходят через центральный узел, принадлежащий частным лицам или компаниям, имеющим над ним полный контроль. Таким образом, подобные системы имеют множество уязвимых мест, от человеческого фактора до возможного отказа или неисправности оборудования. Под децентрализованными подразумеваются системы, в которых отсутствуют узлы, управляющие работой других узлов.

Но такие системы существуют уже давно. Настоящим новшеством являются системы, умеющие достигать децентрализованного консенсуса. До появления подобных решений в любом приложении требовалась некоторая централизация. Теперь же, если приложение требует согласия между участвующими сторонами, то использование цепочки блоков позволяет достичь этого децентрализованным способом. Цепочка блоков - неизменяемая запись, копия которой хранится всеми узлами сети. Если один из участников сети попытается внести в цепочку изменения, не соответствующие консенсусу, то остальные участники отклонят это изменение в автоматическом режиме, сохранив целостность и достоверность информации в цепочке блоков.

Эти нововведения возможно использовать для создания децентрализованных приложений, имеющих высочайшую отказоустойчивость и позволяющих совершать операции между пользователями без третьих лиц.

Это открывает новые возможности для взаимодействия между людьми, а также новые способы решения привычных задач.

Для демонстрации этого целью работы было выделено создание децентрализованного приложения назначения встреч на мероприятиях, в котором всё взаимодействие между пользователями (включая экономическое) происходит без участия третьих лиц.

В рамках работы были выделены следующие задачи:

- провести анализ предметной области технологии блокчейн и выбрать платформу для разработки;
- разработать приложение.

Глава 1. Анализ предметной области технологии блокчейн на примере криптовалюты Bitcoin

Децентрализованные приложения, речь о которых пойдёт в данной работе, работают на основе технологии блокчейн.

Блокчейн — выстроенная по определённым правилам непрерывная последовательная цепочка блоков (связный список), содержащих информацию. Структура цепочки блоков отражена на рисунке 1. Чаще всего копии цепочек блоков хранятся на множестве разных компьютеров независимо друг от друга. Т.к. каждый следующий блок связан с предыдущим, злоумышленники не имеют возможности просто поменять баланс одного из пользователей в следующем блоке. Другие участники сети проверят правильность информации и проигнорируют транзакции злоумышленников, пытающихся несколько раз потратить одни и те же средства или иным образом нарушить работу системы.

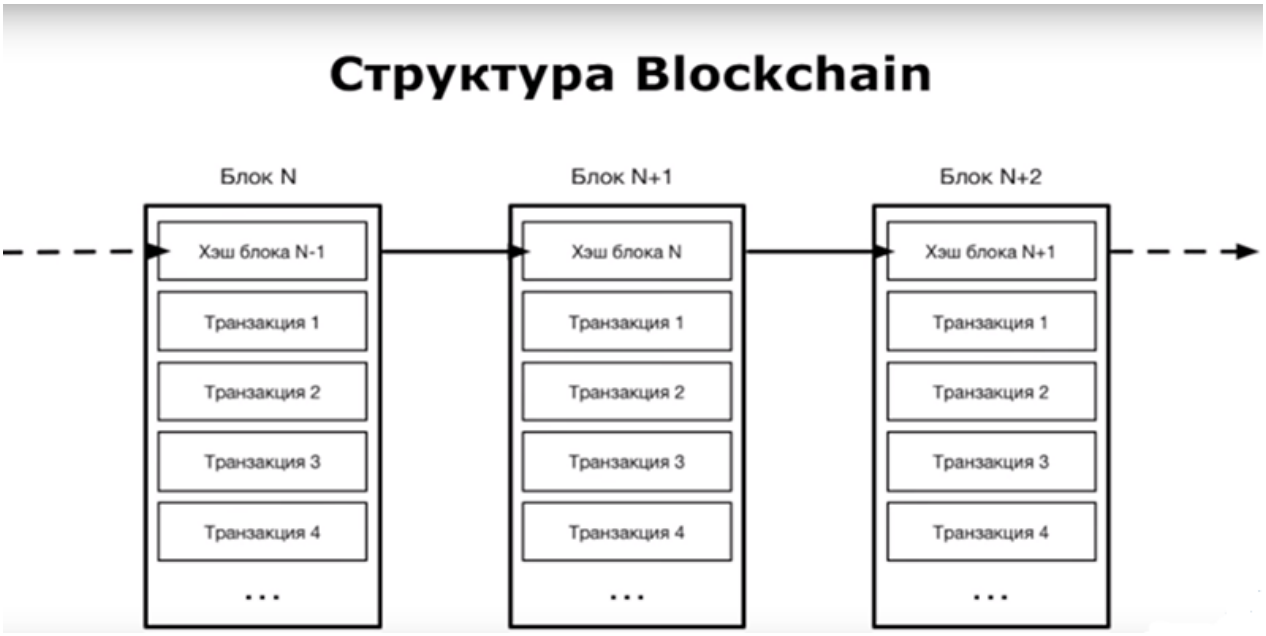


Рисунок 1 - Структура цепочки блоков

Первым применением технологии блокчейн стала распределенная децентрализованной системы - криптовалюта «Биткойн», созданная в 2009

году человеком под псевдонимом Сатоши Накамото. Её появление позволило псевдоанонимно отправлять денежные средства в любую точку планеты, минуя государственные границы [6]. Главной особенностью системы является полная децентрализация.

1.1 Децентрализация

1.1.1 Криптографические основы работы системы Bitcoin

Криптография — это фундаментальная часть Bitcoin. В Bitcoin используется так называемая криптография на эллиптических кривых. Она основана на особой функции — эллиптической кривой [3].

Эллиптическая кривая над полем K —кубическая кривая на проективной плоскости над K ^ (алгебраическим замыканием поля K), задаваемая уравнением 3-й степени с коэффициентами из поля K и «точкой на бесконечности».

Эллиптическая кривая — это функция, как правило, записываемая в виде формы Вейерштрасса:

$$y^2 = x^3 + ax + b$$

В зависимости от значений параметров a и b, график данной функции может выглядеть по-разному. На рисунке 5 представлены несколько примеров таких графиков.

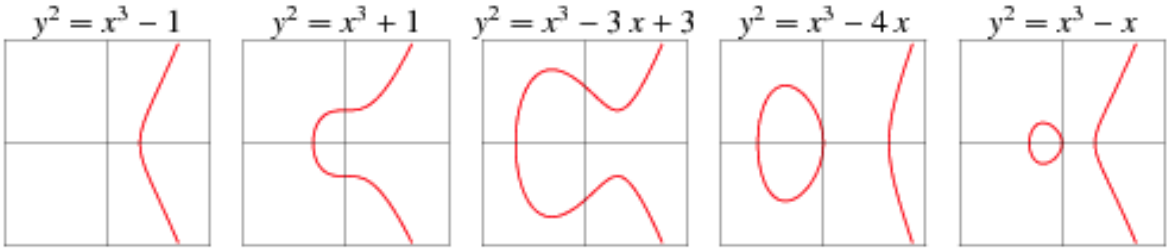


Рисунок 5 – Примеры эллиптических кривых

Пусть есть две точки $P, Q \in \alpha$. Их суммой называется точка $R \in \alpha$, которая в простейшем случае определяется следующим образом: проведем прямую через P и Q — она пересечет кривую α в единственной

точке, назовем ее $-R$. Поменяв Y координату точки $-R$ на противоположную по знаку, получится точка R, которая будет являться суммой P и Q, то есть $P + Q = R$.

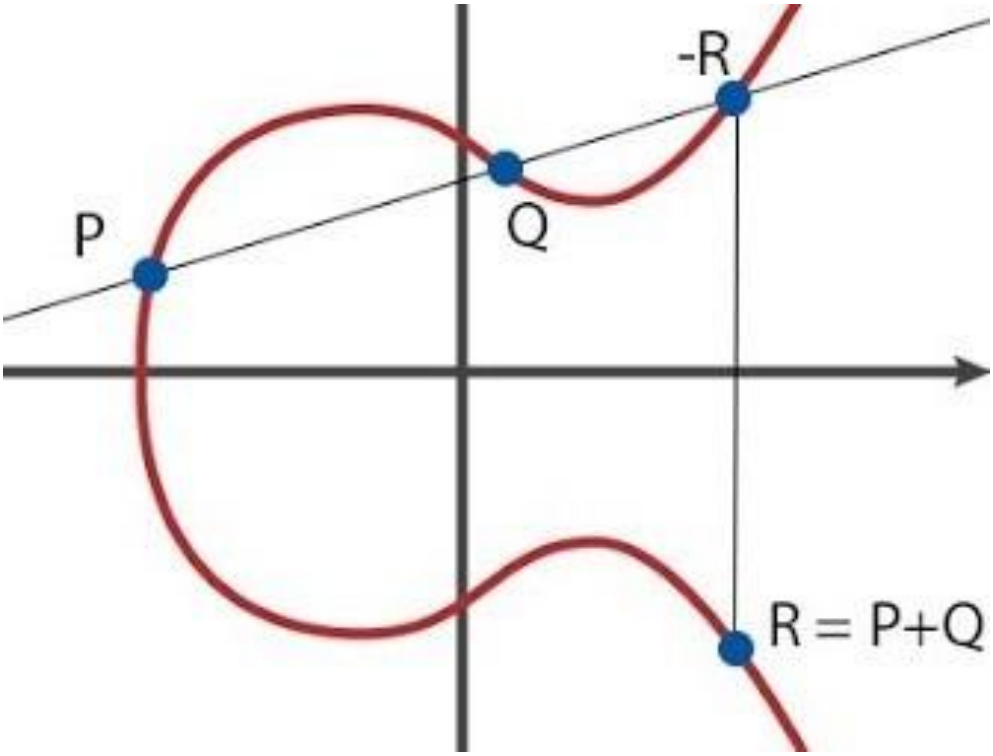


Рисунок 6 – Точки на эллиптической кривой

Важно отметить, что такая операция сложения именно вводится — если складывать точки в привычном понимании, то есть складывая соответствующие координаты, то получится совсем другая точка

$$R'(x_1 + x_2, y_1 + y_2)$$

которая, скорее всего, не имеет ничего общего с R или $-R$ и вообще не лежит на кривой α, как это можно увидеть на рисунке 6.

Возникает вопрос, что произойдёт, если провести прямую через две точки, имеющие координаты вида $P(a, b)$ и $Q(a, -b)$, то есть прямая, проходящая через них, будет параллельна оси ординат (третий кадр на рисунке 7).

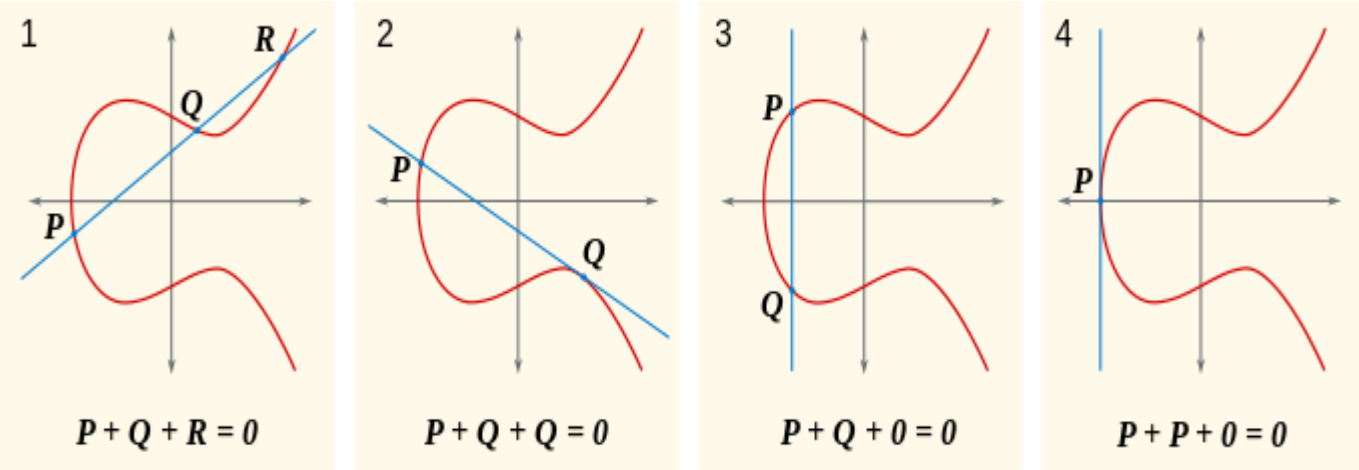


Рисунок 7 – Примеры эллиптических кривых

Как можно заметить, в этом случае отсутствует третье пересечение с кривой α, которое выше называлось –R. Для того, чтобы избежать этого, вводится так называемая точка в бесконечности, обозначенная как O или 0. Таким образом, в случае отсутствия пересечения:

$$P + Q = O.$$

Особый интерес представляет случай, когда необходимо сложить точку саму с собой (2 кадр на рисунке 7, точка Q). В этом случае просто проведем касательную к точке Q и отразим полученную точку пересечения относительно оси y.

Теперь можно ввести операцию умножения точки на число N. В результате получается новая точка

$$K = G * k$$

На рисунке 8 представлена иллюстрация возможных операций с эллиптической кривой.

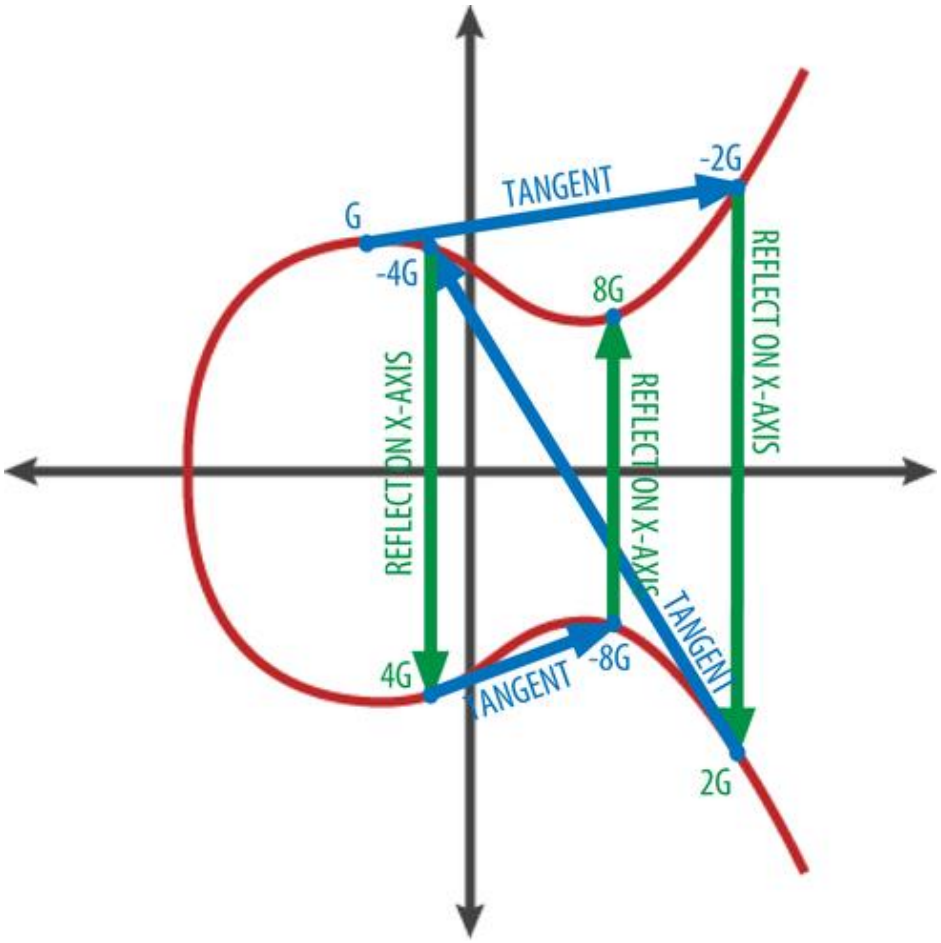


Рисунок 8 – Операции с эллиптической кривой

В эллиптических кривых над конечными полями используется точно такая же кривая, только рассматриваемая над некоторым конечным полем

$$F_p = Z/Z_p = \{0, 1, \dots, p-1\}$$

где p - простое число. Значит

$$y^2 \bmod p = x^3 + ax + b \ (mod\ p)$$

Все названные свойства (сложение, умножение, точка в бесконечности) для такой функции остаются в силе, хотя, если попробовать нарисовать данную функцию, то напоминать привычную эллиптическую кривую она будет лишь отдаленно. А понятие «касательной к функции в точке» вообще теряет всякий смысл, но это не является критичным. Вот пример функции

$$y^2 = x^3 + 7$$

для $p = 17$ (рисунок 9)

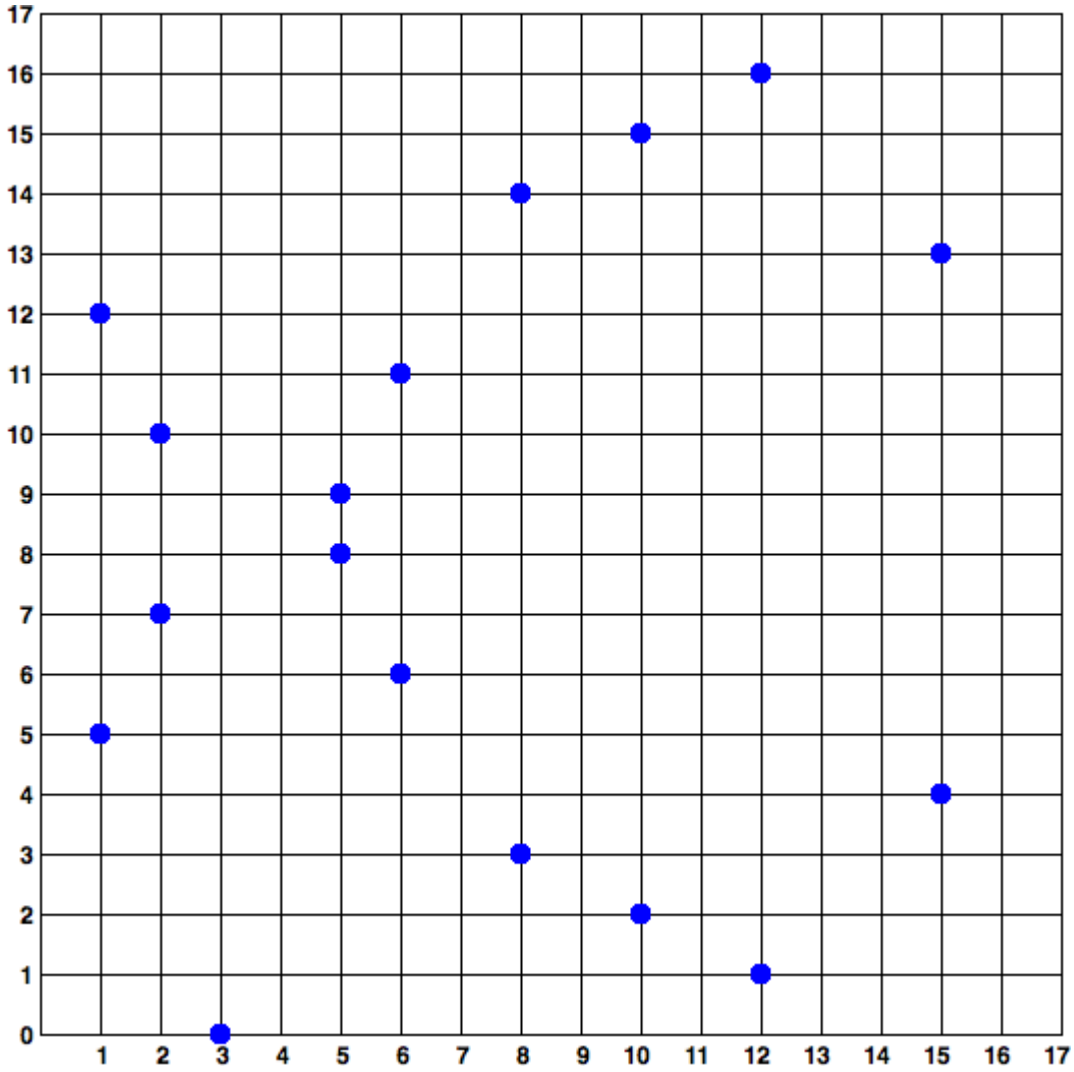


Рисунок 9 – Эллиптическая кривая над конечным полем

В Bitcoin используется кривая под названием SECP256k1. Она имеет вид

$$y^2 = x^3 + 7$$

и рассматривается над полем F_p, где p — очень большое простое число, а именно:

$$2^{256} - 2^{32} - 2^9 - 2^8 - 2^6 - 2^4 - 1 \quad .$$

Так же для SECP256k1 определена так называемая *base point*, она же *generator point* — это просто точка, как правило, обозначаемая G, лежащая на данной кривой. Она нужна для создания публичного ключа, о котором будет рассказано ниже.

Для примера работы системы представляется следующая ситуация: Алиса хочет перевести 1 BTC Бобу. Для этого она создает сообщение:

{

"from" : 1FXySbm7jpJfHEJRjSNPPUqnpRTcSuS8aN, *// Адрес Алисы*

"to" : 1Eqm3z1yu6D4Y1c1LXKqReqo1gvZNrmfvN, *// Адрес Боба*

"amount" : 1 *// Отправить 1 BTC*

}

Потом Алиса берет свой приватный ключ, хэш сообщения и функцию вида

validate_signature(public_key, signature, text)

На выходе она получает подпись своего сообщения. После этого она рассылает всем участникам сети исходное сообщение, подпись и свой публичный ключ.

В результате, каждый участник при желании сможет взять эти данные, функцию вида

validate_signature(public_key, signature, text)

и проверить, действительно ли владелец приватного ключа подписывал это сообщение. А если участники сети знают, что public_key принадлежит Алисе, то можно понять, отправила эти деньги она или же кто-то пытается сделать это от ее имени.

Более того, предположим, что нашелся человек, вставший между Алисой и остальной сетью. Пусть он перехватил сообщение Алисы и что-то в нем изменил. Но даже в этом случае проверка подписи на достоверность покажет, что сообщение было изменено.

Приватный ключ — это общий термин и в различных алгоритмах электронной подписи могут использоваться различные типы приватных ключей.

В Bitcoin используется алгоритм ECDSA. В его случае приватный ключ — это некоторое натуральное 256 битное число, то есть самое обычное целое число от 1 до $2^{256}2$. Технически, даже число *123456* будет являться корректным приватным ключом, но подобные значения очень легко перебираются, и злоумышленник сможет быстро подобрать такой приватный ключ и присвоить себе чужие средства.

Фактически, публичный ключ — это некоторая точка, лежащая на кривой SECP256k1. Он вычисляется по следующей формуле:

$K = G * K$

Где K — приватный ключ;

G — base point.

Операция получения публичного ключа определена однозначно, то есть конкретному приватному ключу всегда соответствует один единственный публичный ключ. Обратная операция является вычислительно трудной и, в общем случае, получить приватный ключ из публичного можно только полным перебором первого.

1.2 Транзакции и формирование блоков

Транзакция – запрос на изменение состояния цепочки блоков. Транзакции похожи на записи в бухгалтерской книге (рисунок 10).

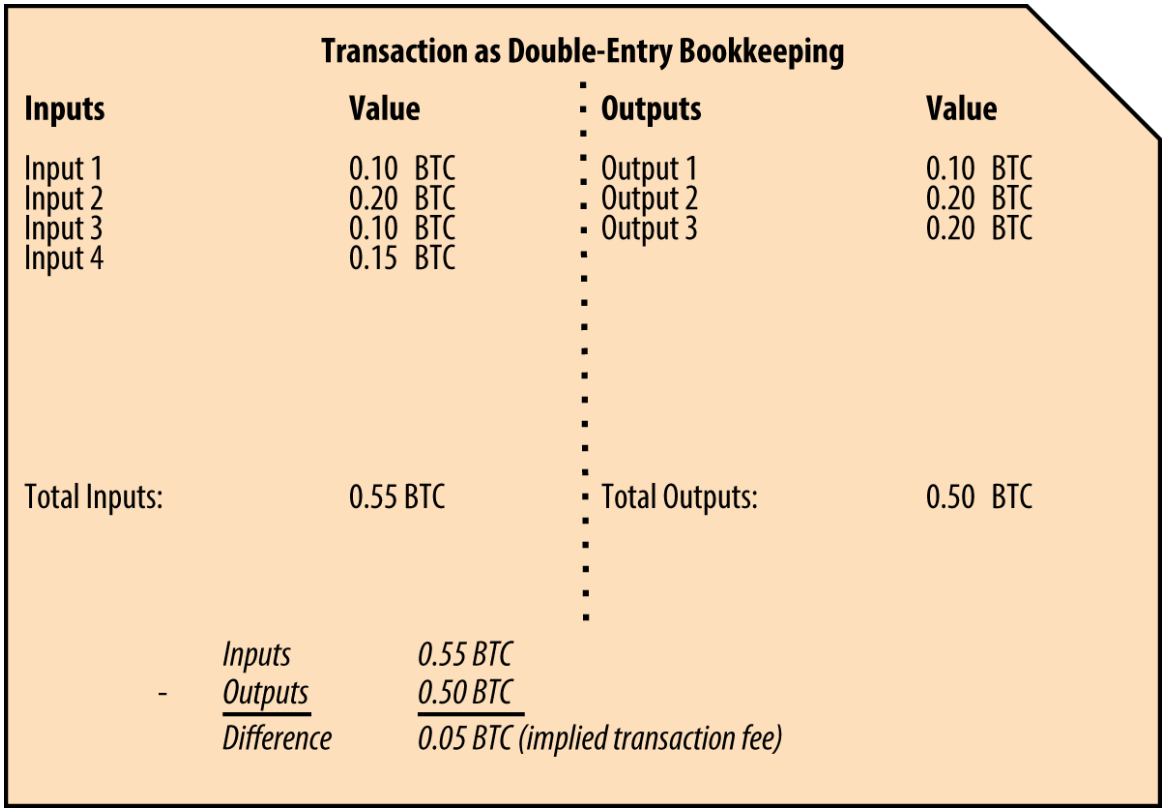


Рисунок 10 – Транзакции в виде записей в бухгалтерской книге

Каждая транзакция содержит один или несколько "входов" (inputs), которые являются дебетом для Bitcoin-аккаунта. С другой стороны транзакции существует один или несколько "выходов" (outputs), которые являются кредитом для Bitcoin-аккаунта (рисунок 11).

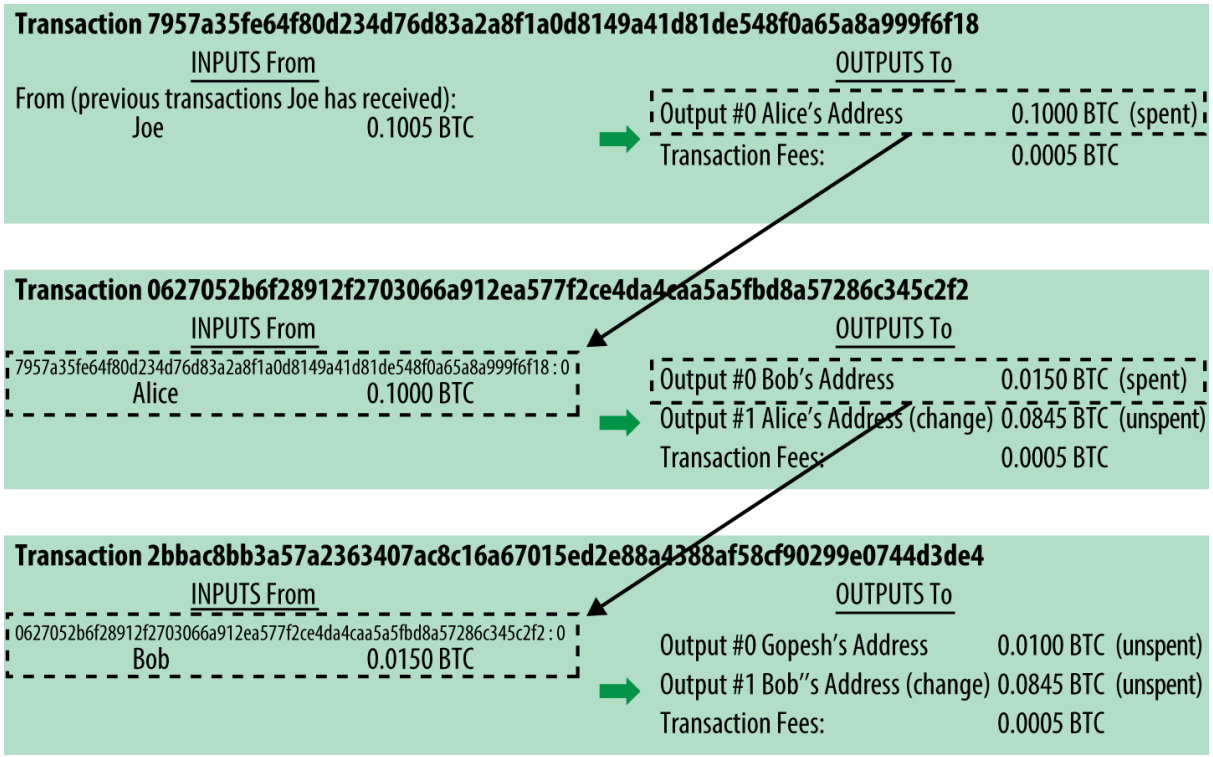


Рисунок 11 – Транзакции в виде записей в бухгалтерской книге

Входы и выходы (дебеты и кредиты) не обязательно должны складываться в одну сумму. Вместо этого, выходы зачастую создаются чуть меньшими чем входы, а разница между ними представляет собой комиссию за транзакцию, которая является оплатой так называемым майнерам, вносящим её в блок.

Транзакция также содержит доказательство владения для каждого количества биткоинов (входов), которые будут потрачены, в форме цифровой подписи владельца, которую может проверить любой участник сети. В терминологии Bitcoin и других криптовалют "потратить" - это подписать криптографической подписью транзакцию, которая пересылает валюту новому владельцу, идентифицируемому по публичному адресу. Информация о транзакциях внутри сети передаётся в формате json, пример такой транзакции отражён на рисунке 12.

Цепочка транзакций, где выход одной транзакции — это вход следующей транзакции:


```
{
  "version": 1,
  "locktime": 0,
  "vin": [
    {
      "txid":
"7957a35fe64f80d234d76d83a2a8f1a0d8149a41d81de548f0a65a8a999f6f18",
      "vout": 0,
      "scriptSig" :
"3045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24c
b02204b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e3813
[ALL]
0484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc541233637
6789d172787ec3457eee41c04f4938de5cc17b4a10fa336a8d752adf",
      "sequence": 4294967295
    }
  ],
  "vout": [
    {
      "value": 0.01500000,
      "scriptPubKey": "OP_DUP OP_HASH160
ab68025513c3dbd2f7b92a94e0581f5d50f654e7 OP_EQUALVERIFY OP_CHECKSIG"
    },
    {
      "value": 0.08450000,
      "scriptPubKey": "OP_DUP OP_HASH160
7f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a8 OP_EQUALVERIFY OP_CHECKSIG",
    }
  ]
}
```


Рисунок 12 – Пример транзакции в формате json

1.3 Особенности использования технологии блокчейн

Значительным недостатком технологии блокчейн является низкая пропускная способность. Для работы технологии тысячи компьютеров должны постоянно обмениваться информацией об изменении состояния блокчейна. Для достижения согласия о том, какая информация будет внесена в следующий блок, необходим алгоритм консенсуса. Одним из таких алгоритмов является «доказательство выполнения работы», когда клиент для проведения операции в системе должен решить криптографическую задачу. В большинстве криптовалют для достижения этой цели используется майнинг.

Майнинг — деятельность по созданию новых структур (обычно речь идёт о новых блоках в блокчейне) для обеспечения функционирования криптовалютных платформ [9]. За создание очередной структурной единицы обычно предусмотрено вознаграждение за счёт новых (эмитированных) единиц криптовалюты и/или комиссионных сборов. Обычно майнинг сводится к серии вычислений с перебором параметров для нахождения хеша с заданными свойствами. В биткойн в процессе майнинга новый блок находится примерно каждые 15 минут.

Всё это приводит к тому, что пропускная способность блокчейна биткойн примерно равна 7 транзакциям в секунду. При этом, примерное энергопотребление системы по разным оценкам колеблется от 3 до 5 гигаватт.

Из всего вышеперечисленного можно сделать вывод, что на данном этапе своего развития технология блокчейн является очень медленной по сравнению с централизованными решениями, способными проводить сотни тысяч транзакций в секунду с гораздо меньшим энергопотреблением, но это компенсируется полной её децентрализованностью.

1.4 Возможности развития технологии блокчейн

Со временем стало очевидно, что технологию блокчейн можно использовать не только для создания электронных валют. Начали появляться другие варианты технологии. Некоторые из них предлагали заменить используемую повсеместно систему DNS, другие хранить различные данные в определённом виде. В 2013 году канадский программист российского происхождения Виталик Бутерин предложил концепцию «биткойн 2.0». Основной его особенностью стали смарт-контракты.

Смарт-контракт - это электронный алгоритм или условие, при выполнении которого стороны могут обмениваться деньгами, недвижимостью, акциями и другими активами. Для реализации умного контракта нужно иметь децентрализованную сеть, где все участники имеют равные права. В качестве финансового инструмента используется криптовалюта.[6]

Иными словами, это небольшие программы, хранящиеся в блокчейне и выполняющиеся в распределённой сети на нескольких компьютерах одновременно. Идея нашла практическое решение уже в 2015 году с выходом платформы Ethereum, что сделало возможным создание децентрализованных приложений.

1.5 Появление децентрализованных приложений

Появление смарт-контрактов позволило реализовать децентрализованные приложения, т.к. с их помощью можно реализовать любую логику в распределённой сети. Таким образом у людей впервые появилась возможность не только пересылать друг-другу денежные средства через интернет без участия третьих лиц, но и устанавливать сложные экономические взаимоотношения. Смарт-контракты стали использоваться для сбора средств на реализацию проекта, создания децентрализованных организаций, управление которыми происходит на основе логики смарт-контрактов и т.д.

Стало возможным создавать “неостанавливаемые” приложения, не имеющие точек отказа и доступные без каких-либо перерывов, связанных с обновлениями/переездом в другой дата-центр и другими причинами, по которым централизованные сервисы могут стать недоступны.

1.6 Пример приложения на основе технологии блокчейн

Наглядным примером реализации децентрализованного приложения на платформе Ethereum является децентрализованная криптовалютная биржа EtherDelta (рисунок 13).

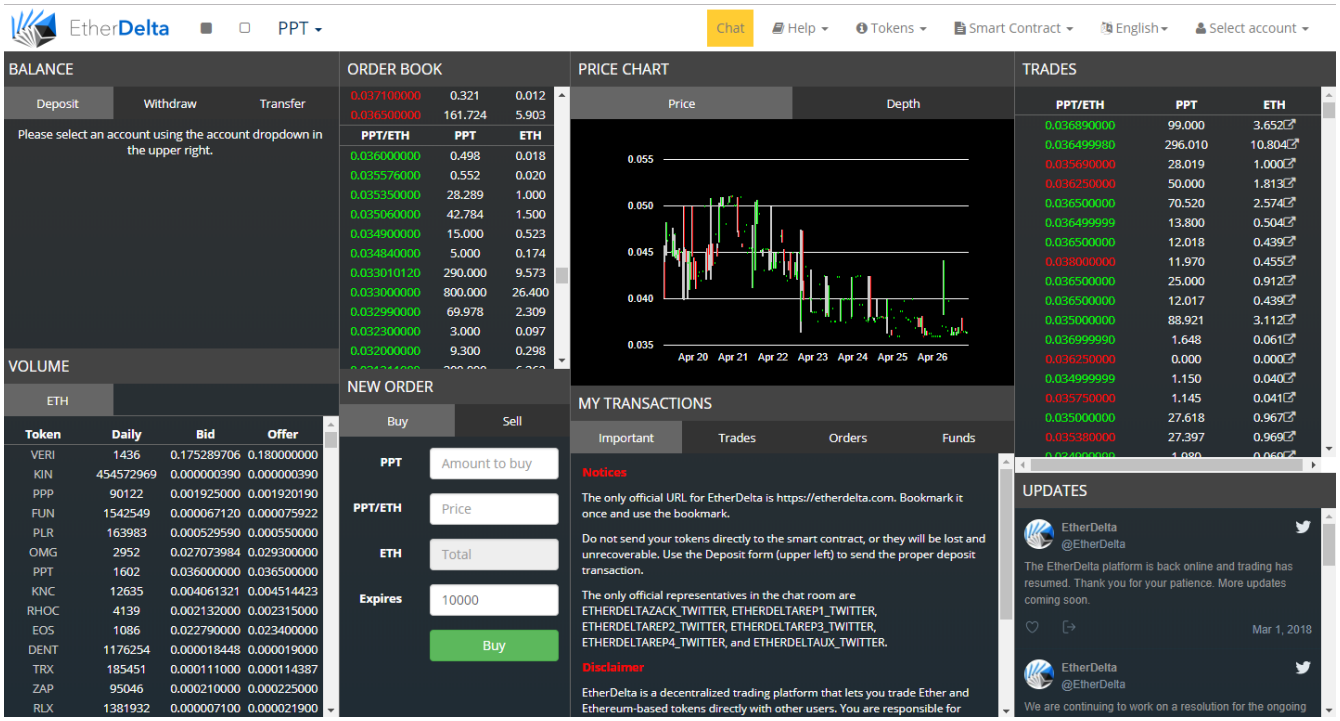


Рисунок 13 – Интерфейс криптовалютной биржи EtherDelta

Для начала работы с биржей необходимо создать новый аккаунт. Для этого необходимо выбрать пункт Select account – New account (Рисунок 14)

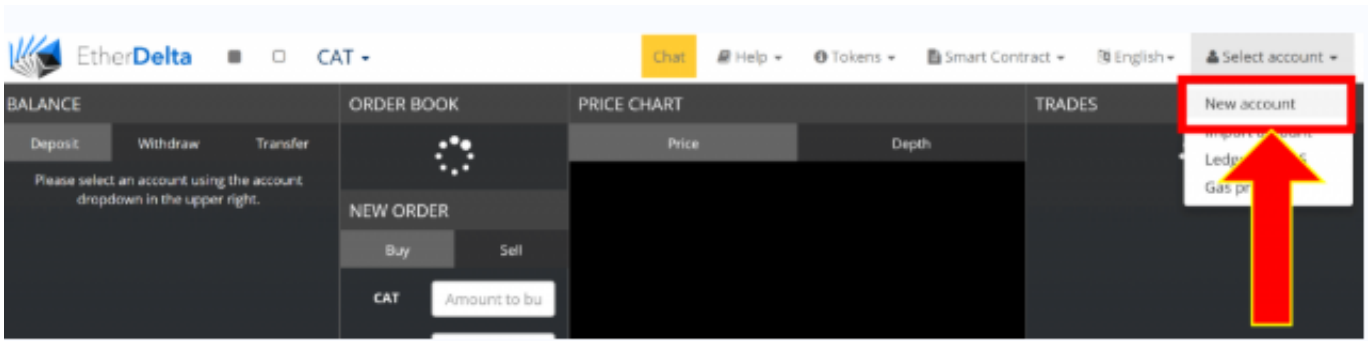


Рисунок 14 – Интерфейс создания аккаунта EtherDelta

После этого появится всплывающее окно, содержащее публичный и приватный ключи пользователя (рисунок 15).

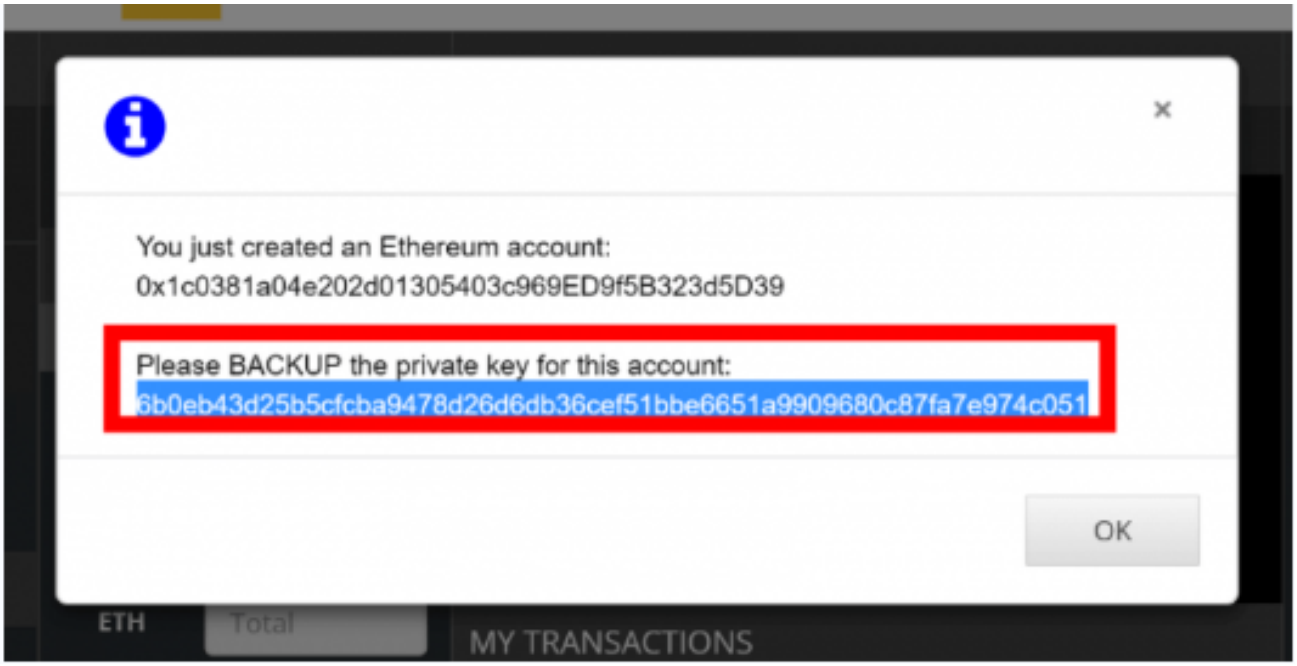


Рисунок 15 – Всплывающее окно с приватным и публичным ключами

Для защиты от злоумышленников необходимо хранить свой приватный ключ в безопасном месте и не распространять его. Публичный же ключ можно предоставлять другим участникам сети, если необходимо провести различные операции.

Затем для совершения операций в сети необходимо пополнить баланс. После успешного пополнения баланса можно выбрать токен, которым пользователь собирается торговать (Рисунок 16).

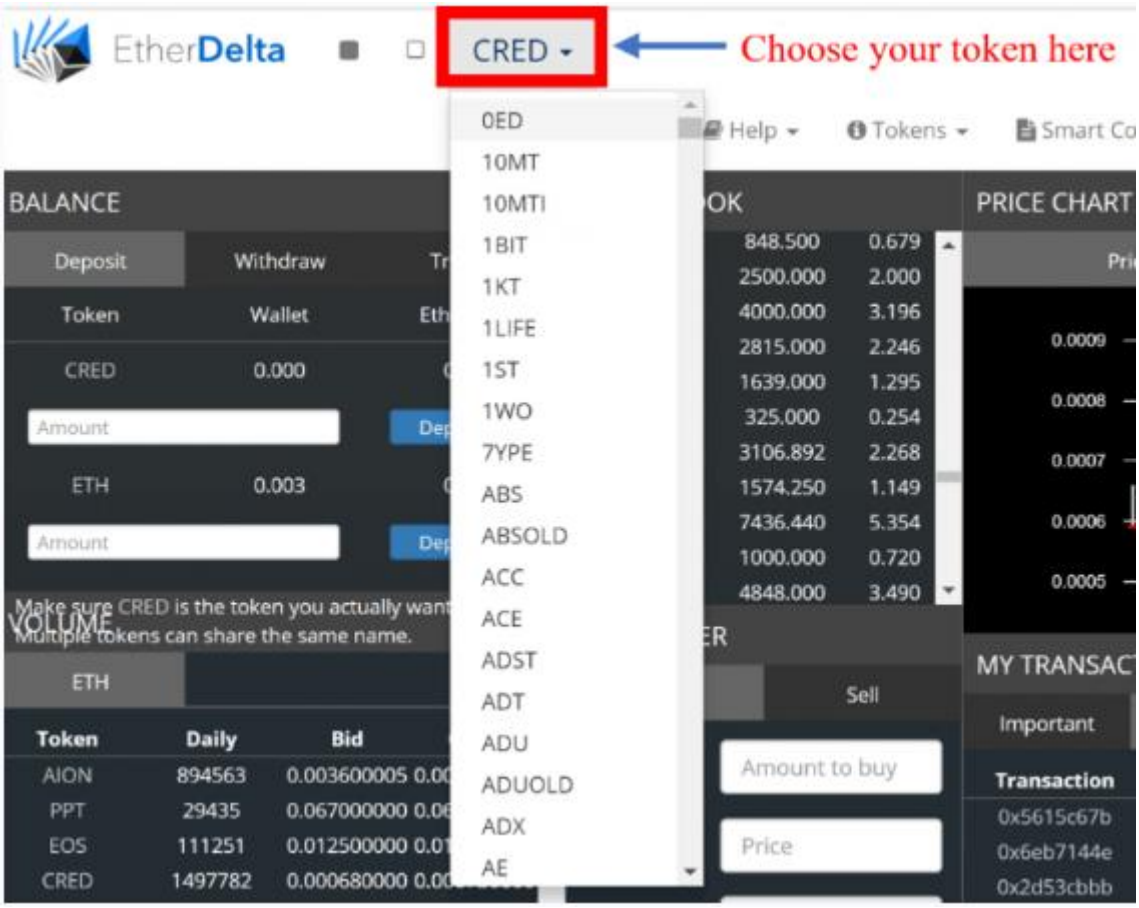


Рисунок 16 – Интерфейс выбора токенов

Токены – монеты, хранящиеся в цепочке блоков Ethereum. Пользователи могут обмениваться ими в рамках сети, а также использовать как экономическую основу децентрализованных приложений. Биржа EtherDelta в текущем виде не позволяет торговать монетами не использующими сеть Ethereum в основе своей работы.

Затем происходит основная часть взаимодействия приложения с пользователем. Так как EtherDelta является децентрализованной биржей, в её работе используются смарт-контракты. На рисунке 17 представлен интерфейс взаимодействия со смарт-контрактом.

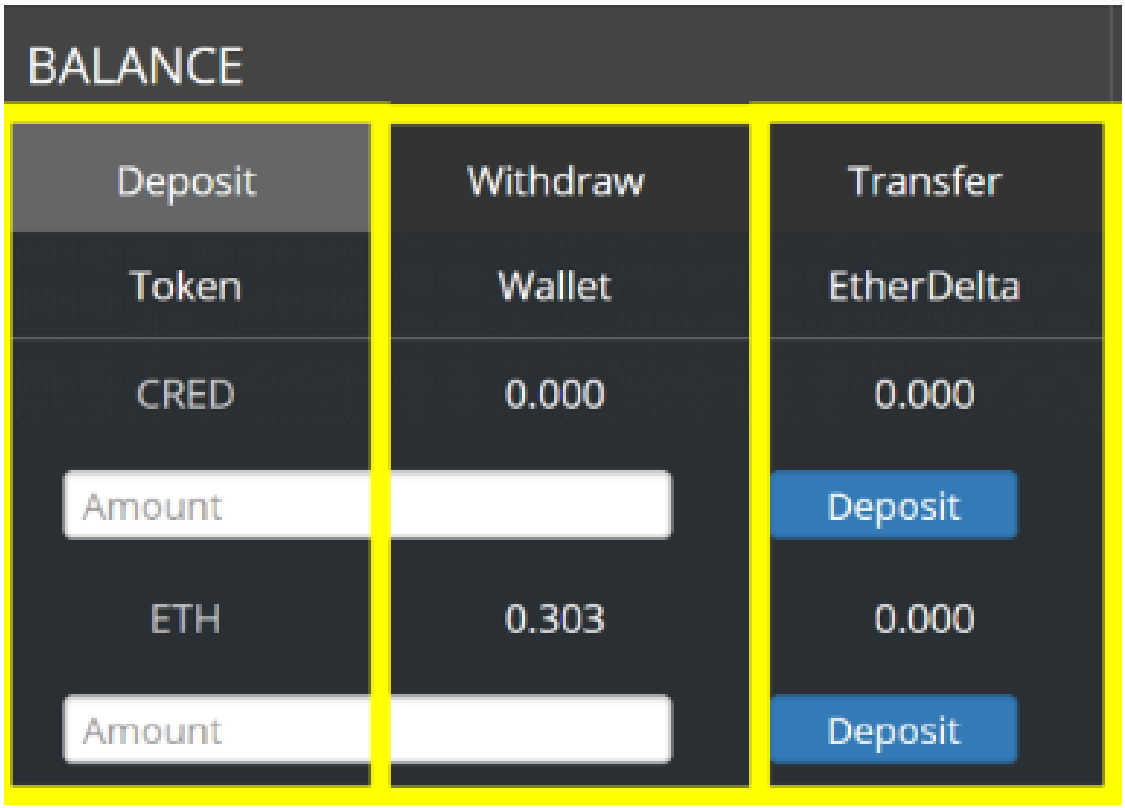


Рисунок 17 – Интерфейс выбора токенов

Этот интерфейс поделён на три основные группы:

- Token (токен): в этом поле указан символ токена торгующегося в рамках данного контракта и монета против которой он торгуется;
- Wallet (кошелёк): в этом поле отражается баланс на кошельке пользователя;
- EtherDelta: смарт-контракт, позволяющий устанавливать ордера на покупку и продажу.

Вся логика данного приложения хранится и выполняется в децентрализованной сети Ethereum, а каждый участник может посмотреть исходный код проекта и решить для себя, стоит ли ему доверять. При этом исходный код смарт-контракта имеет объём всего лишь в 312 строк кода. Главной особенностью проекта является то, что владельцы биржи не управляют средствами пользователей, это исключает возможность обмана пользователей организаторами [15].

1.7 Выводы по результатам первой главы

В данной главе проведён анализ предметной области блокчейн. Были выведены и перечислены особенности работы технологии, основы для её появления, технологии и техники лежащие в основе её работы. Приведен наглядный пример работы децентрализованного приложения криптовалютной биржи. По итогам данной главы выбраны основные технологии для реализации децентрализованного приложения назначения встреч на мероприятиях.

Глава 2. Разработка приложения

2.1.1 Основная платформа разработки

Основной для разработанного децентрализованного приложения является платформа Ethereum.

Ethereum – децентрализованная платформа разработки, которая позволяет выполнять смарт-контракты. Это позволяет создавать приложения без таких недостатков как цензура, простой, мошенничество или вмешательство третьей стороны.

Эти приложения выполняются на цепочке блоков, представляющей из себя мощную глобальную сеть, способную осуществлять передачу ценностей и отражать владение собственностью.

Это позволяет разработчикам создавать рынки, записывать данные о дебитах и кредитах, отправлять средства согласно записанным инструкциям без участия третьих лиц.

Обменные единицы Ethereum называются эфиром (англ. ether). Для обозначения используется сокращение ETH и символ в виде Ξ (греческая буква Кси). Дробные части имеют свои названия: 1/1000 — finney, 1/106 — szabo, 1/1018 — wei [14].

В отличие от других криптовалют, авторы не ограничивают роль эфира платежами, а предлагают его, например, в качестве средства для обмена ресурсами или регистрации сделок с активами при помощи умных контрактов, в частности авторы назвали эфир «криптотопливом» для исполнения умных контрактов одноранговой сетью. Эфир продаётся на сервисах по обмену, а также добывается при обслуживании распределённой сети (майнинг).

Данная платформа даёт возможность регистрации любых сделок с любыми активами, что делает её конкуретной по отношению к

имеющимся системам регистрации сделок. По мнению некоторых экспертов, технология смарт-контрактов (умных контрактов) знаменует собой новую эру в финансовых технологиях.

Умные контракты в Ethereum представляют собой классы, которые могут быть реализованы на различных языках и компилируются в байт-код для виртуальной машины Ethereum (Ethereum Virtual Machine, EVM) перед отправкой в блокчейн.

Виртуальная машина Ethereum имеет поддержку циклов, поэтому платформа использует механизм, именуемый газом или горючим для ограничения контрактов, которые могут занять много времени для выполнения.

Блокчейн Ethereum является, по сути, системой состояния транзакций. В информатике такое понятие, как «система состояний» или «машина состояний» – это система, которая обрабатывает вводимую информацию и на основании последней преобразуется в новое состояние

Одним из важных моментов в системе Ethereum является процесс оплаты. За любое вычисление, осуществляющееся в результате проведения операций с транзакциями внутри сети Ethereum, берется определенная плата. Номинал данной оплаты носит название «горючее» (от англ. gas).

Горючее – это единица измерения, которая используется для определения размера оплаты по конкретному вычислению. Цена на горючее – это количество «эфира», которое вы способны потратить на каждую единицу горючего. Измеряется цена на горючее в «gwei». Wei является самой маленькой единицей эфира, где 10^18 Wei –это всего 1 эфир. Один gwei равен 1 000 000 000 Wei.

Для проведения любой транзакции отправитель должен установить лимит горючего, а также цену на горючее. Цена на горючее и лимит

горючего – это максимальная сумма в Wei, которую отправитель готов заплатить за проведение транзакции.

Важным аспектом работы Ethereum является то, что любая операция, которая выполняется сетью, также одновременно выполняется каждым полным узлом. Таким образом, все шаги, связанные с вычислением на виртуальной машине Ethereum, стоят значительных денежных средств. Для решения простых задач (например, запуска простой бизнес-логики, проверки подписей, а также иных операций, связанных с криптовалютой) смарт-контракты Ethereum являются отличным решением. Но в случаях, когда требуется выполнение других, более сложных, задач: хранение файлов, выполнение задач из области машинного обучения, которые могут вызывать чрезмерную загрузку сети, стоимость проведения операций является слишком высокой. Введение оплаты предотвращает действия пользователей, направленные на излишнюю загрузку сети.

2.1.2 Основной язык разработки

В качестве основного языка программирования для разработки смарт-контрактов был выбран язык программирования Solidity.

Solidity является высокоуровневым объектно-ориентированным, предметно-ориентированным языком программирования смарт-контрактов для платформы Ethereum. В языке используется статическая типизация. Программы на языке Solidity компилируются в байт-код для Ethereum Virtual Machine (EVM) – виртуальной машины Ethereum [15].

Имеет схожий с языком ECMAScript (JavaScript) синтаксис. По замыслу создателей языка, это должно помочь принятию языка и быстрой адаптации к нему существующих веб-разработчиками. Однако, в отличие от ECMAScript, Solidity получил статическую типизацию переменных и динамические типы возвращаемых значений. По сравнению с другими компилируемыми в такой же байт код виртуально машины Ethereum языками Serpent и Mutan язык Solidity имеет важные отличия. Поддерживаются комплексные переменные контрактов, включая произвольные иерархические отображения (mappings) и структуры. Контракты поддерживают наследование, включая множественное и C3-линеаризацию. Поддерживается бинарный интерфейс программирования (ABI), имеющий множество типобезопасных функций в каждом контракте (в последствии появился также и в Serpent). Специфицирована система документирования кода для пользовательского пояснения последовательности вызовов, получившая название «Спецификации на естественном языке Ethereum» (Ethereum Natural Specification Format)

Solidity всё ещё находится на ранней стадии разработки и постоянно дорабатывается

На данный момент этот язык является стандартом для разработки смарт-контрактов на платформе Ethereum, несмотря на наличие аналогов.

2.1.3 Основная программная платформа разработки

Для упрощения разработки и взаимодействия с сетью Ethereum была использована программная платформа Truffle Framework.

Функциональные возможности Truffle:

- Компиляция смарт-контрактов, их объединение и управление ими;
- Автоматизация тестирования контрактов при помощи Mocha и Chai;
- Изменяемая цепочка построения приложений, позволяющая вносить в процесс значительные изменения;
- Платформа для автоматизации процесса миграции контрактов и развёртывания приложений;
- Интерактивный консольный интерфейс;
- Мгновенная рекомпиляция элементов проекта в процессе разработки;
- Возможность использования скриптов для работы с окружением Truffle.

Truffle значительно ускоряет процесс разработки и позволяет автоматизировать большинство задач, связанных с разработкой смарт-контрактов [2].

2.1.4 Эмулятор сети Ethereum

В основной сети Ethereum за каждую произведённую операцию (транзакцию) необходимо оплатить комиссию майнерам, поддерживающим инфраструктуру сети, для чего используется эфир (Ether), который необходимо приобрести за реальные деньги или получить за вклад в работу сети (майнинг). Более того, пропускная способность сети ограничена и при максимальной нагрузке достигает лишь 14 транзакций в секунду. Из этого можно сделать вывод, что основная сеть Ethereum не подходит для использования в процессе разработки, где часто необходимо изменять программный код и размещать его в сети.

Для этих задач используются эмуляторы сети Ethereum. Они полностью копируют поведение сети Ethereum (за исключением децентрализованного консенсуса), используя при этом меньше ресурсов и имея более высокую производительность.

В рамках работы был выбран эмулятор под названием Ganache. При каждом запуске им создаётся новая цепочка блоков с несколькими адресами, имеющими на своих счетах значительное количество эфира. Это позволяет проводить тестирование работы взаимодействия между пользователями как просто в сети путём передачи ценностей друг-другу, так и при тестировании смарт-контрактов.

Для каждого адреса можно увидеть:

- публичный ключ;
- баланс (по умолчанию 100 ETH);
- количество совершённых транзакций;
- индекс;
- приватный ключ.

Таким образом можно использовать при разработке все возможности платформы Ethereum без её недостатков в виде относительно высокой стоимости операций и медленного их выполнения.

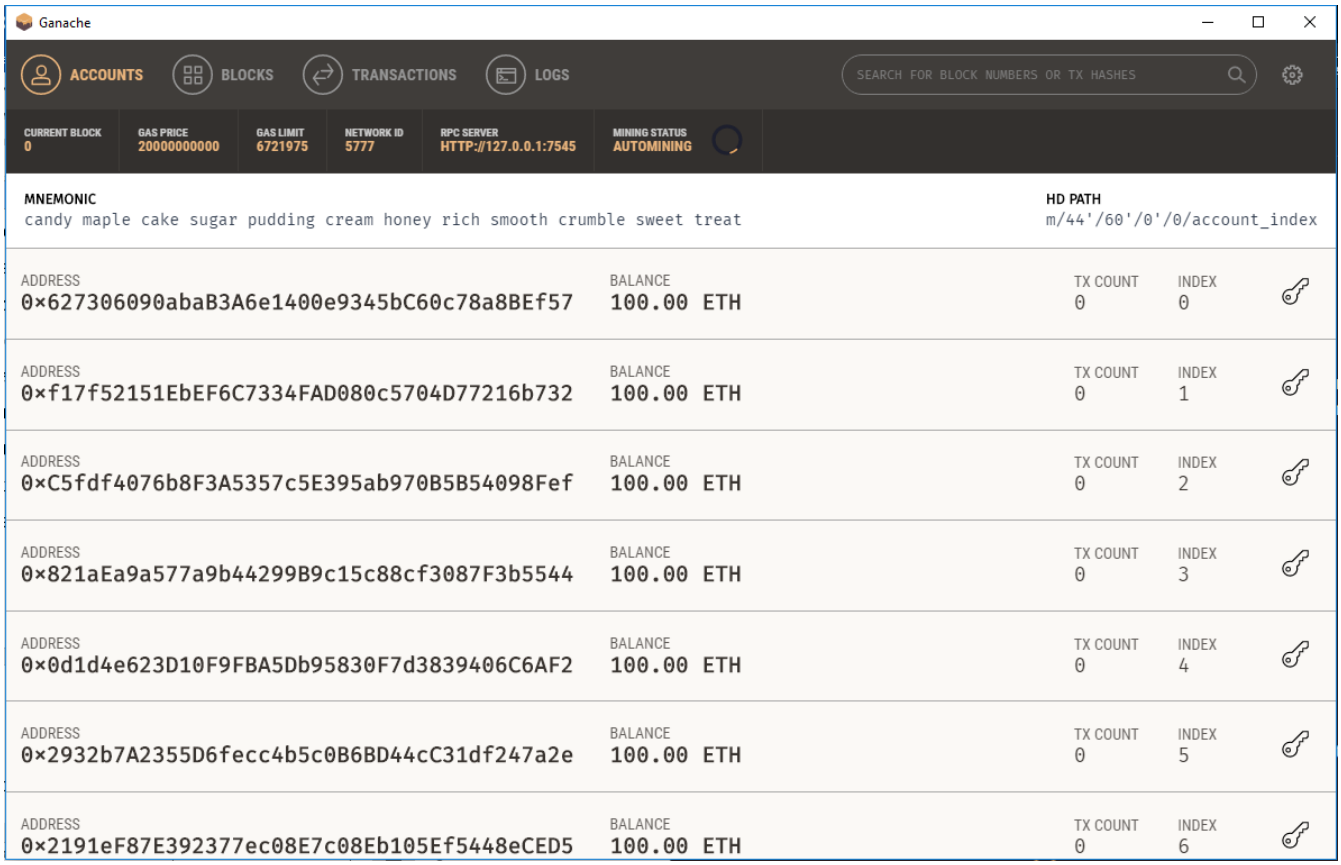


Рисунок 14 – Интерфейс эмулятора сети Ethereum

2.1.5 Клиент для взаимодействия с сетью Ethereum

Для работы с сетью Ethereum необходимо иметь клиент, способный с ней взаймодействовать. Т.к. частью проекта является веб-приложение, было принято решение использовать дополнение для браузера под названием MetaMask.

Плагин выполняет роль промежуточного звена между основным кошельком и обычными сайтами. На него переводится ETH и другие токены, необходимые пользователю, а все остальные активы желательно хранить на ином кошельке, менее подверженному взлому.[10]

Это делается, чтобы защитить данные криптовалютных кошельков, а также, чтобы сделать работу с ETH и другими токенами удобной и быстрой.

Кроме этого плагин предоставляет доступ к сети Ethereum через свои узлы. Важным фактором является то, что он позволяет установившему его клиенту полноценно пользоваться библиотекой web3js для взаимодействия с сетью Ethereum. Web3js является собранием модулей, включающих в себя различный функционал сети Ethereum. В рамках работы использовался модуль web3-eth, предназначенный для работы с цепочкой блоков и смарт-контрактами.

2.1.6 Стек веб-разработки

Важной частью приложения является веб-интерфейс, позволяющий пользователю взаимодействовать со смарт-контрактом. Основой для веб-приложения является платформа NodeJS

Node.js — программная платформа, основанная на движке V8 (транслирующем JavaScript в машинный код), превращающая JavaScript из узкоспециализированного языка в язык общего назначения.[5] В рамках работы она используется как веб-сервер, выдающий необходимые файлы по запросу от браузера. (Серверная часть приложения)

На стороне клиента используется библиотека Vue — это прогрессивная библиотека для создания пользовательских интерфейсов. Она состоит из различных элементов, которые можно использовать по отдельности. Её ядро в первую очередь решает задачи уровня представления (view), что упрощает интеграцию с другими библиотеками и существующими проектами. С другой стороны, Vue полностью подходит и для создания сложных одностраничных приложений (SPA, Single-Page Applications), если использовать её совместно с современными инструментами и дополнениями.

Для управления состоянием веб-интерфейса используется VueX - паттерн управления состоянием и библиотека для приложений на Vue. Она включает в себя хранилище, являющееся единственным источником правдивости информации для приложения. Таким образом можно манипулировать используемыми в приложении данными и предсказуемо представлять их.

Это работает следующим образом: компоненту (component) во время появления на экране необходима информация, он использует действие (action) для получения необходимых данных. После этого выполняется асинхронный вызов программного интерфейса (API) с запросом этих

данных. Когда данные получены, действие передаст эти данные в мутацию (mutation). Затем мутация изменяет состояние хранилища, после чего компонент перерисовывается.

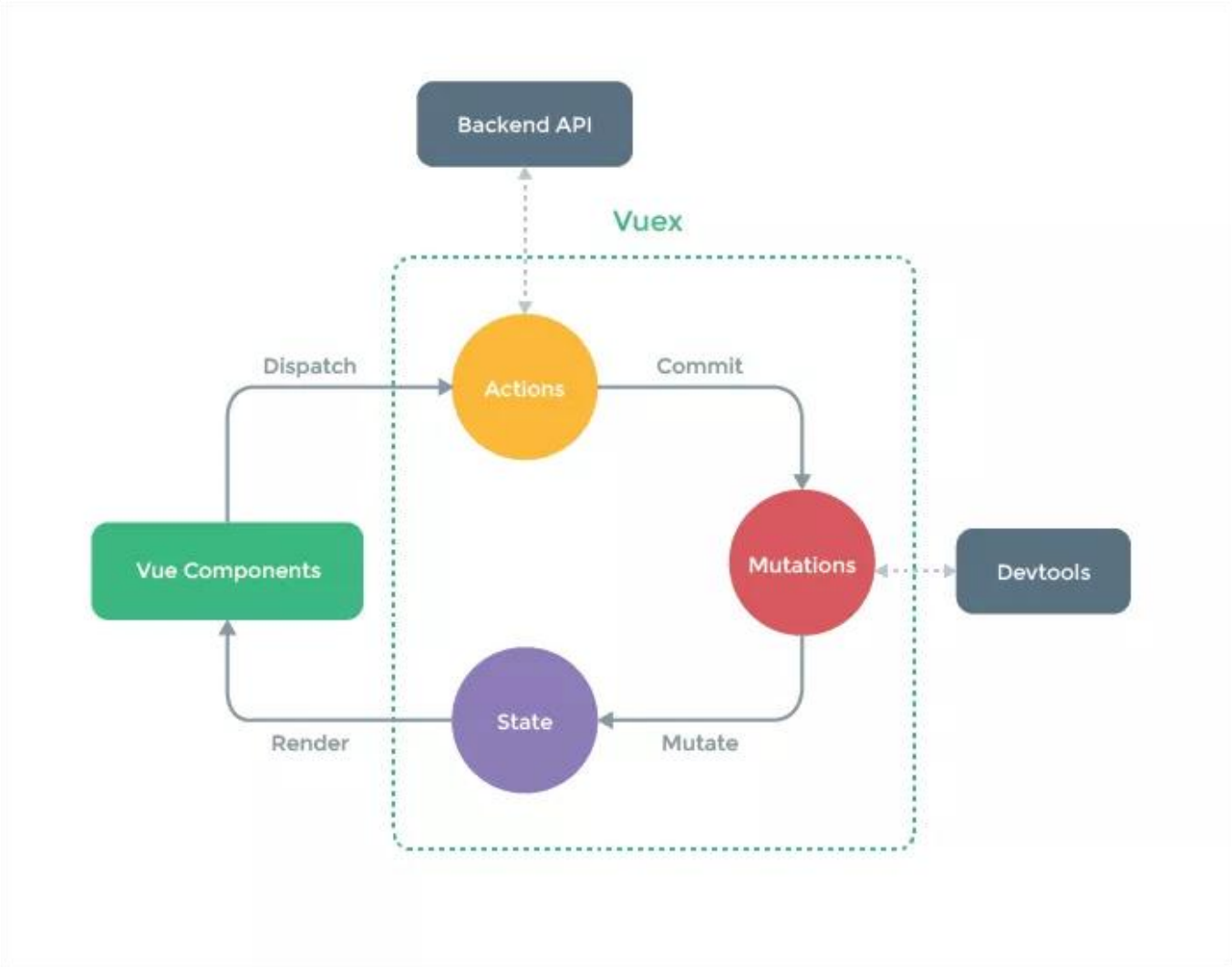


Рисунок 15 – Архитектура клиентской части приложения

2.2 Разработка программы

2.2.1 Логика смарт-контракта

Вся логика приложения хранится и выполняется в распределённой сети. Важно правильно спроектировать смарт-контракт, чтобы по результату его выполнения получался однозначныfй результат. На рисунках 16 и 17 представлена логика смарт-контракта в виде SADT-модели

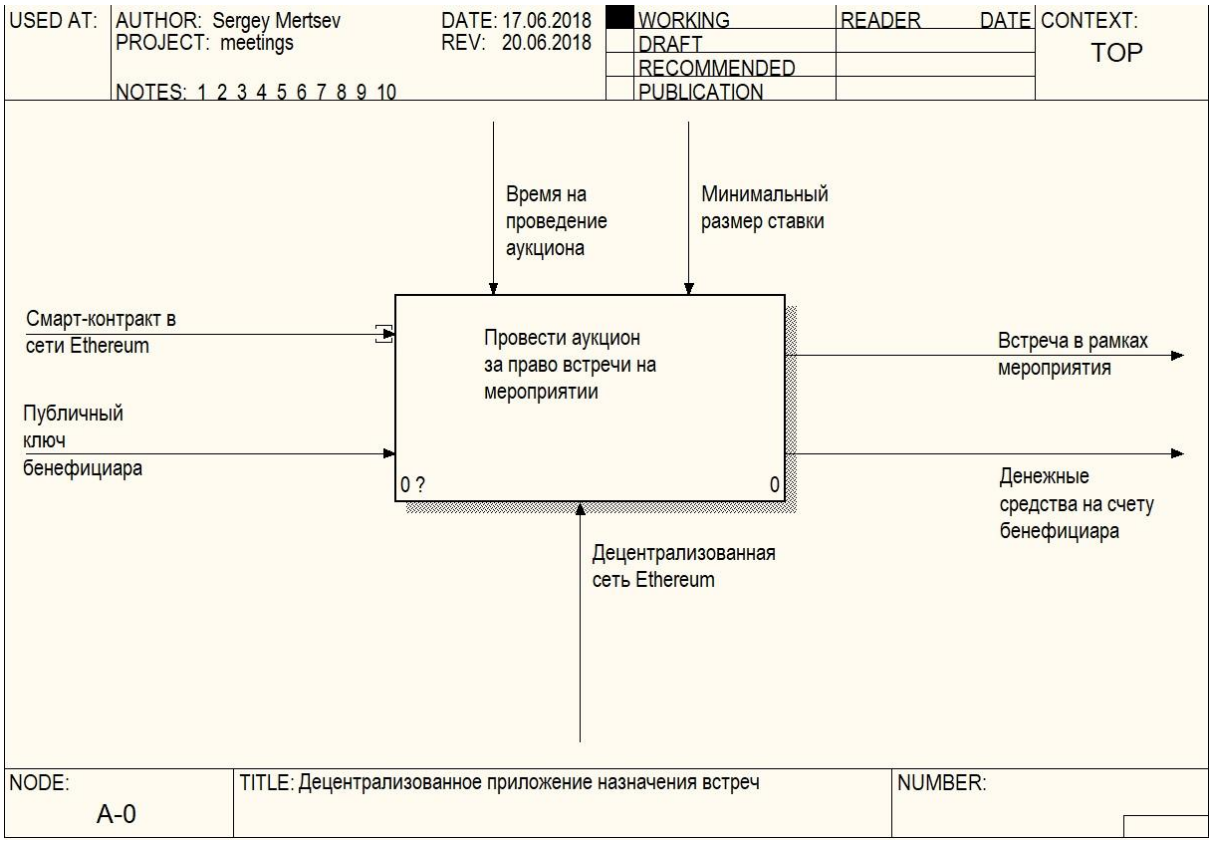


Рисунок 16 – SADT-модель смарт-контракта аукциона

На рисунке 17 подробно отражены этапы проведения аукциона, от создания контракта и указания в нём параметров проведения, до контроля исполнения условий распределенной сетью и завершение аукциона с последующей передачей средств бенефициару.

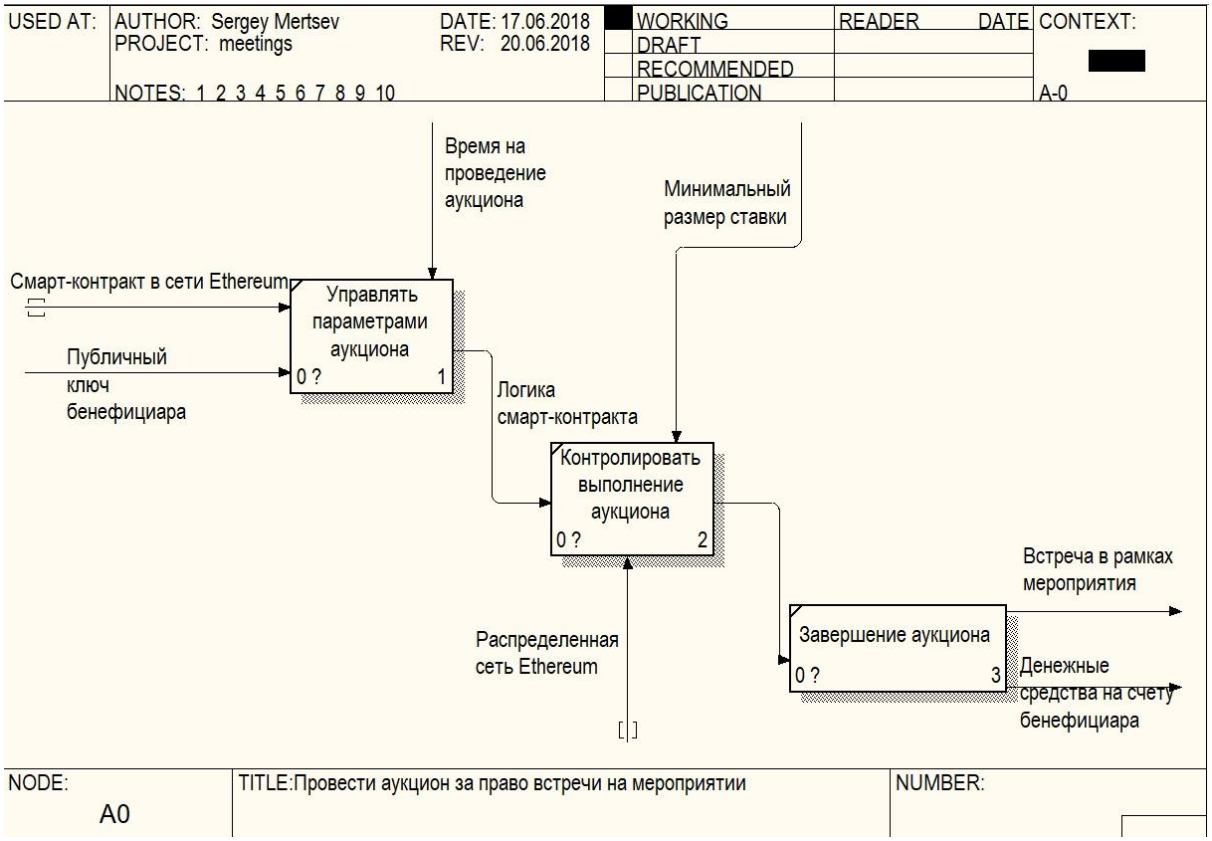


Рисунок 17 – SADT-модель смарт-контракта аукциона

2.2.2 Подробности реализации

Как было указано выше, для взаимодействия с сетью Ethereum необходимо установить расширение для бразуера MetaMask. После установки и создания кошелька оно будет иметь вид как на рисунке 16:

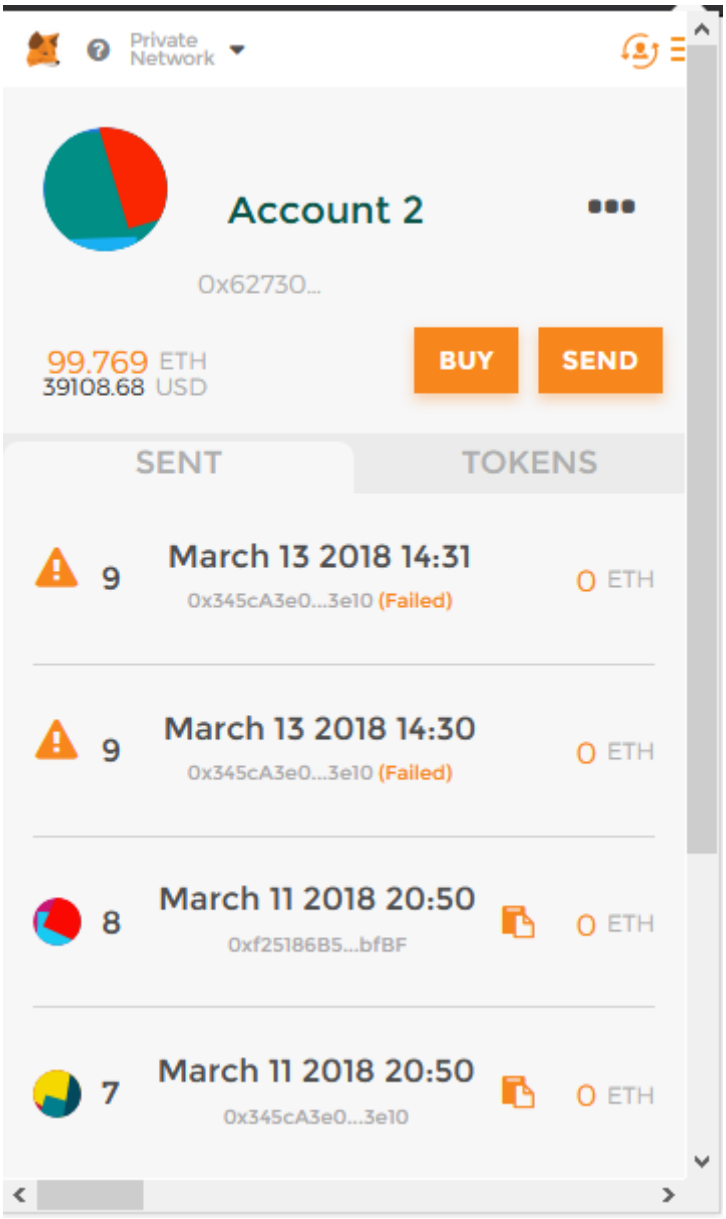


Рисунок 16 – интерфейс расширения MetaMask

В случае если MetaMask не будет установлен, приложение не сможет работать с библиотекой web3js и получать информацию из распределённой сети, поэтому было принято решение не показывать

интерфейс приложения, если оно ещё не установило связь с клиентом Ethereum.

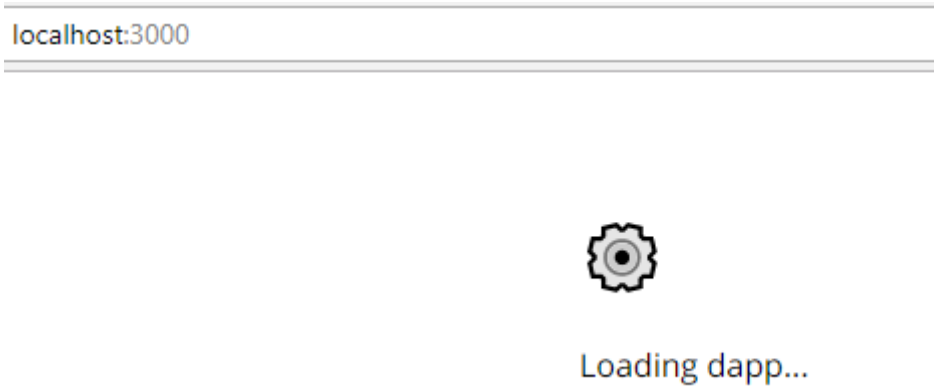


Рисунок 17 – Интерфейс приложения при отсутствии подключения к сети Ethereum

Данный веб-интерфейс отображает информацию об аккаунте пользователя и состоянии смарт-контракта аукциона (рисунок 18).

Рисунок 18 – Интерфейс веб-приложения

Подразумевается, что организатор аукциона уже разместил смарт-контракт в сети ethereum. В дальнейшем планируется добавить возможность размещать смарт-контракты в сети Ethereum “одной кнопкой”.

2.3 Выводы по результатам второй главы

В данной главе была проведена разработка и реализация смарт-контракта и работающего совместно с ним веб-интерфейса. Были перечислены компоненты системы, описаны принципы и особенности их работы и приведены доводы, объясняющие выбор данных компонентов для использования в рамках проекта. По итогу главы реализован минимально жизнеспособный продукт, который уже можно использовать для организации экономических взаимоотношений между людьми.

Заключение

В развитии интернета сформировался явный тренд к появлению децентрализованных приложений и сервисов. Это позволит смягчить недостатки централизованного интернета, имеющиеся на данном этапе.

В связи с этим необходимо изучать возможности современных платформ для реализации децентрализованных приложений и использовать их в повседневной жизни, делая различные экономические процессы более эффективными и менее затратными.

В работе были рассмотрены современные средства для разработки децентрализованных приложений, описаны достоинства и недостатки.

Был разработан алгоритм работы децентрализованного приложения для организации встреч на мероприятиях и выполнена его реализация.

Разработанное приложение может быть использовано для создания конкурентного и открытого экономического сотрудничества и организации встреч экономически заинтересованных участников, а так же для продажи билетов с хранением права собственности на них в цепочке блоков.

Список сокращений

1. ETH – Ethereum
2. BTC – Bitcoin
3. И.с - информационная система, б.д. и т.д., аис.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Как работает Эфириум (Ethereum)? [Электронный ресурс] // Habrahabr. – Режим доступа: https://geektimes.ru/post/294611/
2. Truffle Suite - Your Ethereum Swiss Army Knife [Электронный ресурс] // Truffle Official Website. – Режим доступа: http://truffleframework.com
3. Андреас Антонопулос. Mastering Bitcoin: Programming the Open Blockchain 2nd Edition – Себастопол, Калифорния, США: O'Reilly Media, 2017. – 40-85 с.
4. Андреас Антонопулос. Mastering Ethereum: Building Smart Contracts and Dapps – Себастопол, Калифорния, США: O'Reilly Media, 2017. – 30-90 с.
5. Docs: Node.Js [Электронный ресурс] // Node.Js Official Website. – Режим доступа: https://nodejs.org/en/docs/
6. Криптовалюта как новый инструмент денежного рынка : доклад, тезисы доклада. / О. В. Старова, М. С. Фирскина. –Красноярск Сибирский федеральный университет. –Красноярск, 2017. -148 с.
7. Гуляев Г.Ю. Современная экономика: Актуальные вопросы, достижения и инновации // Москва, 2017. – С. 148-151.
8. Развитие Биткоина в России : научное издание / Б. Б. Османов // Постулат. - Биробиджан: Федеральное государственное бюджетное образовательное учреждение высшего образования Приамурский государственный университет имени Шолом-Алейхема, 2017. т.№5-1.-С. 66
9. Трубникова Е. И. Феномен криптовалюты: характеристики, предпосылки, институциональный анализ рынка / Е. И. Трубникова // Инфокоммуникационные технологии, 2014. т.Т. 12,N № 3.-С.90-94

10. Цой В. В. Обеспечение безопасности при использовании криптовалюты / В. В. Цой, Е. О. Царев, Ю. Е. Домбровский // Банковское дело, 2017,N № 11.-С.78-82
11. Архипов А. Мастерчейн - доверенная среда для всех / А. Архипов // Банковское обозрение, 2017,N № 6.-С.49
12. АНАЛИЗ ОСНОВНЫХ ФАКТОРОВ ФОРМИРОВАНИЯ КУРСА БИТКОЙНА : доклад, тезисы доклада / В. И. Райк. - 2017 // Инновационное развитие современной науки: проблемы, закономерности, перспективы:"Наука и Просвещение" (ИП Гуляев Г.Ю.), 2017. т.Часть 2.-С.48-51
13. Свон, М. Блокчейн. Схема новой экономики : перевод с английского / М. Свон. Москва [науч. ред. В. Фомин]. – 2016
14. Ethereum – Blockchain App Platform [Электронный ресурс] // Ethereum Project. – Режим доступа: https://ethereum.org/
15. Solidity – Solidity 0.4.21 Documentation [Электронный ресурс] // Read the docs. – Режим доступа: https://solidity.readthedocs.io/en/v0.4.21/
16. СТО 4.2–07–2014 Система менеджмента качества. Общие требования к построению, изложению и оформлению документов учебной деятельности. – Введ. 30.12.2013. – Красноярск : ИПК СФУ, 2009. – 60 с.

ПРИЛОЖЕНИЕ А

Плакаты презентации

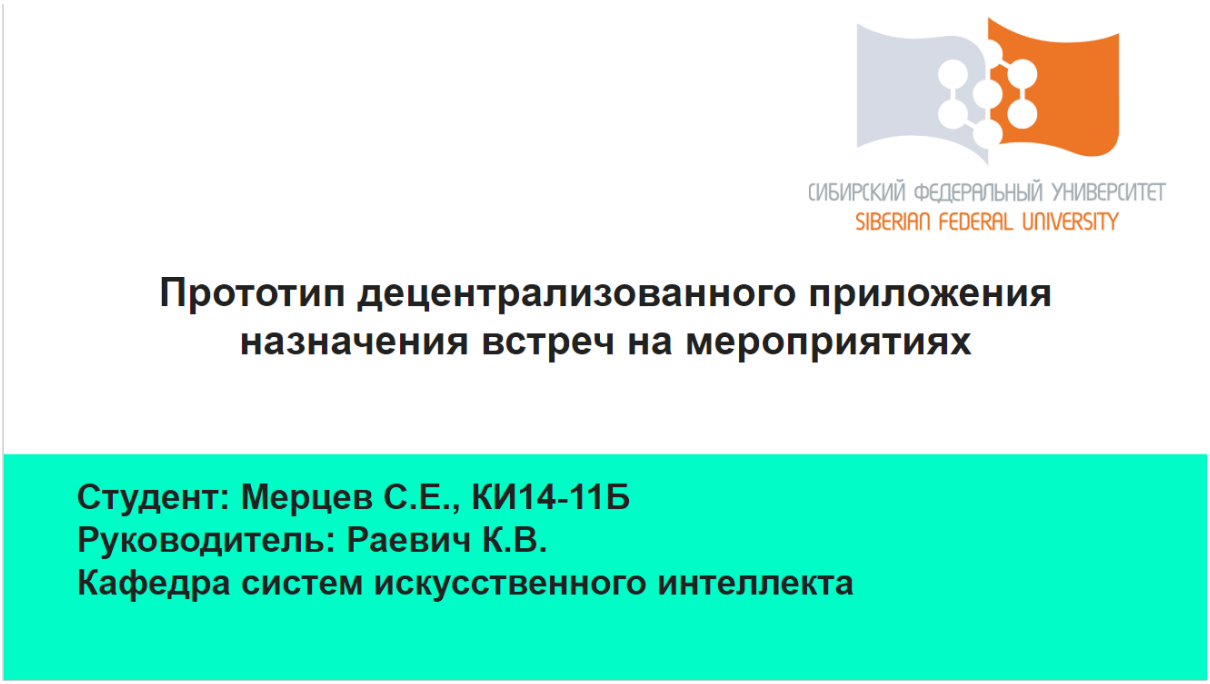


Рисунок А.1 - Плакат презентации 1

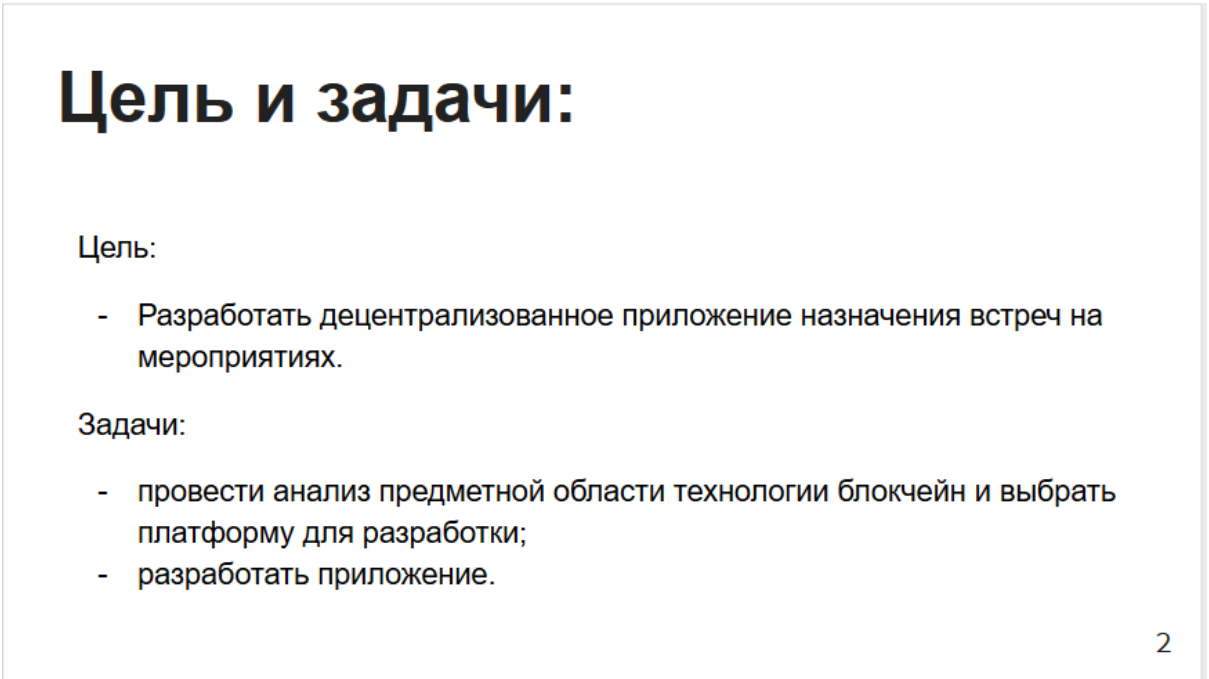
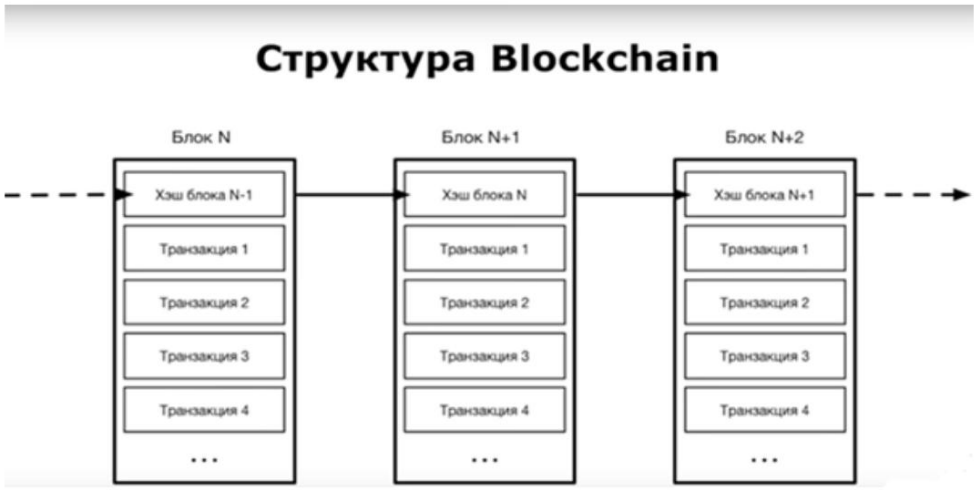


Рисунок А.2 - Плакат презентации 2

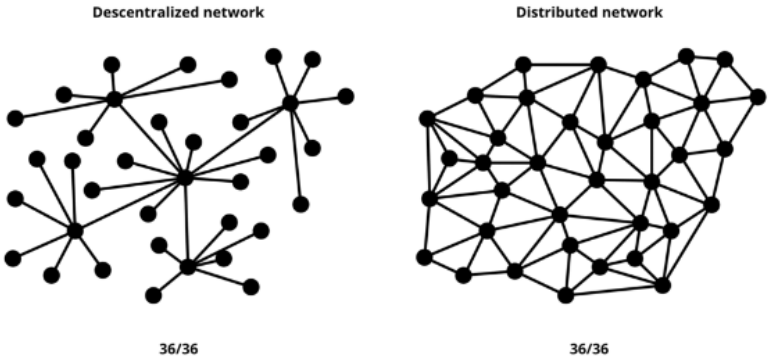
Цепочка блоков



3

Рисунок А.3 - Плакат презентации 3

Распределенные компьютерные сети



4

Рисунок А.4 - Плакат презентации 4

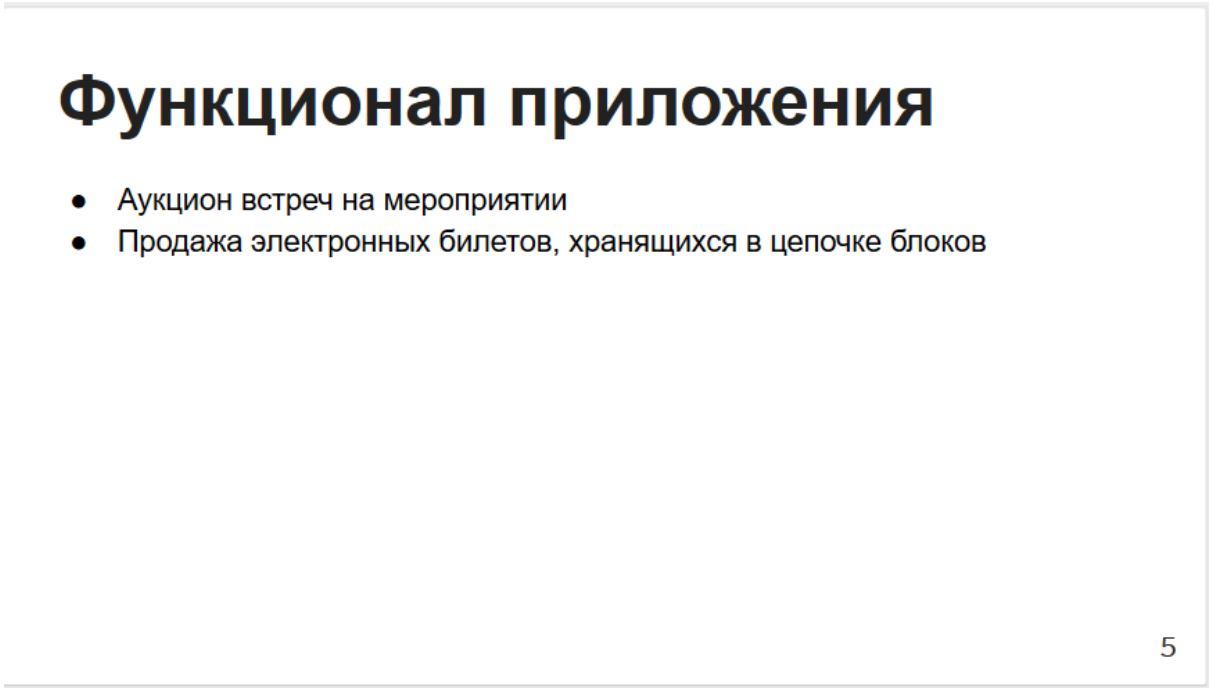


Рисунок А.5 - Плакат презентации 5

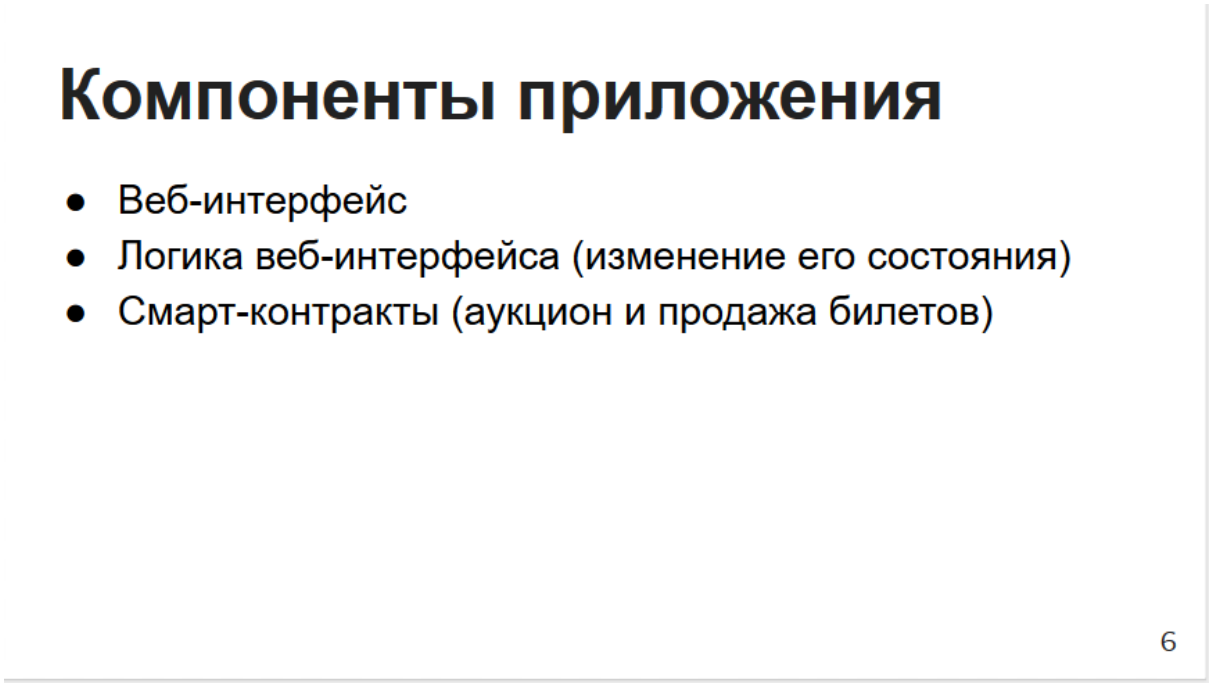
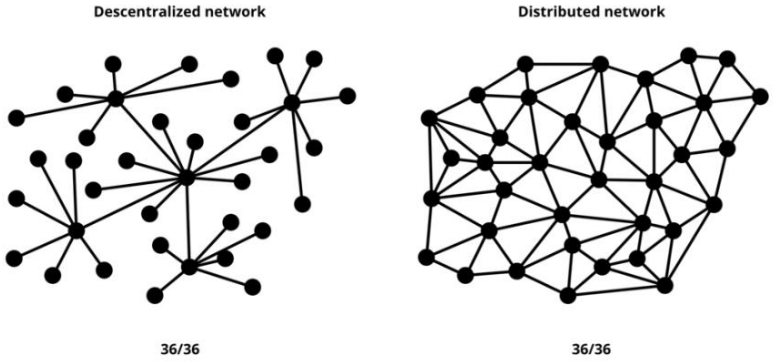


Рисунок А.6 - Плакат презентации 6

Распределенные компьютерные сети



7

Рисунок А.7 - Плакат презентации 7

ВЫБОР ПЛАТФОРМЫ РАЗРАБОТКИ

Требования:

- Децентрализованность (отсутствие контролирующего органа)
- Открытость (открыты исходный код и возможность разработки)
- Надежность (платформа стабильно работает в течение длительного времени)
- Удобство разработки и наличие документации
- Распространенность (значительное количество людей, использующих платформу)

7

Рисунок А.8 - Плакат презентации 8

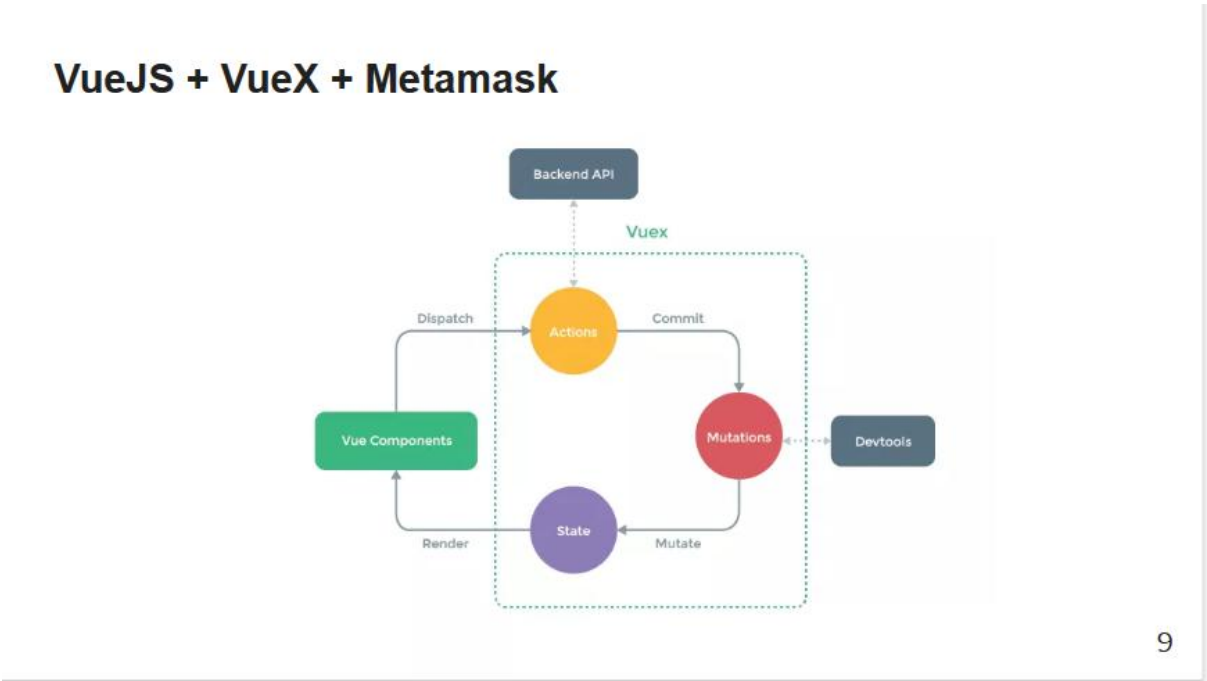


Рисунок А.9 - Плакат презентации 9

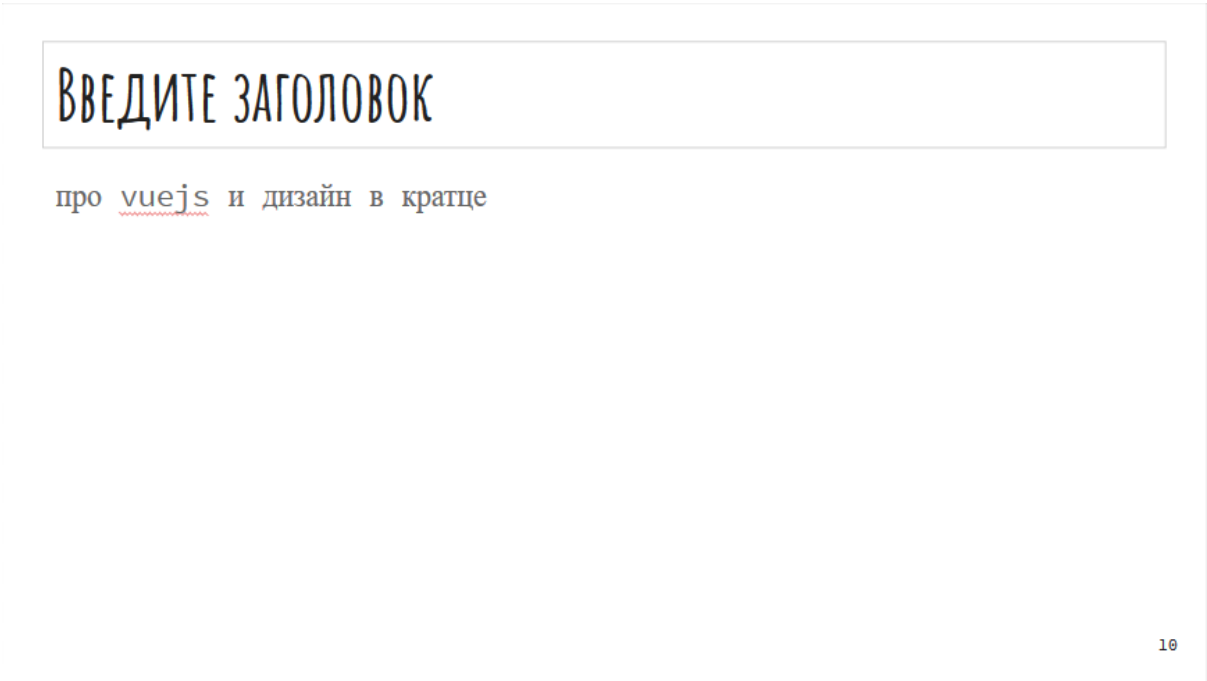


Рисунок А.10 - Плакат презентации 10

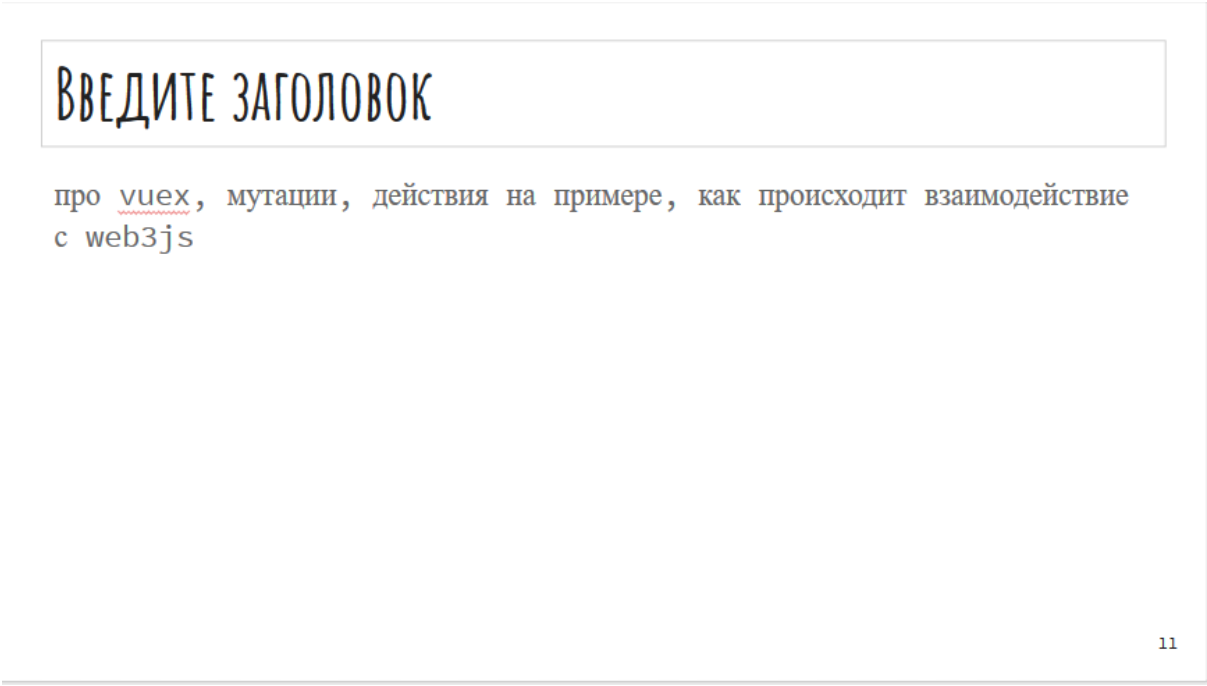


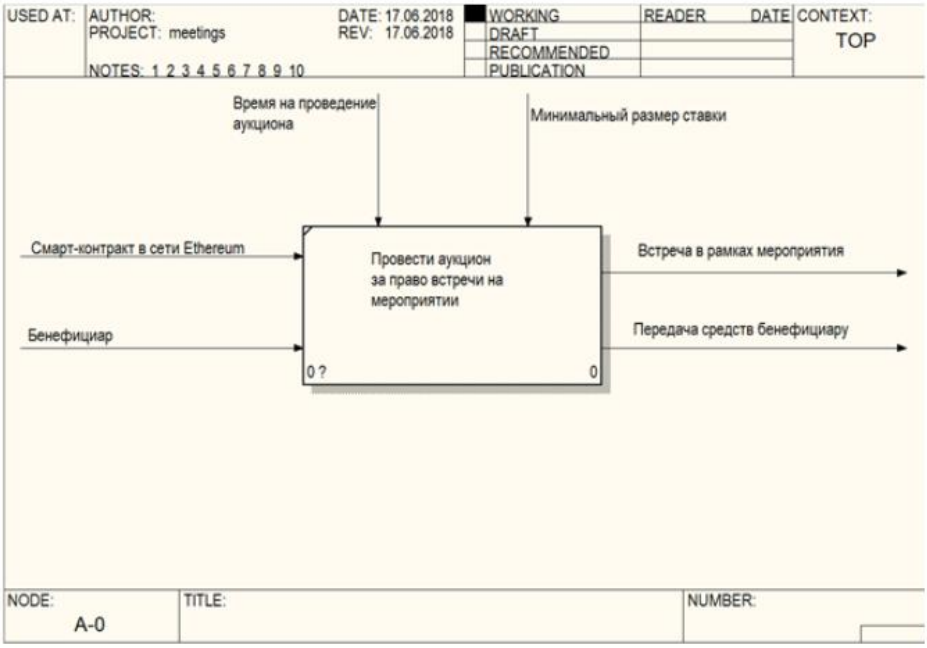
Рисунок А.11 - Плакат презентации 11

Введите заголовок

про смарт-контракты, какие в них реализованы методы

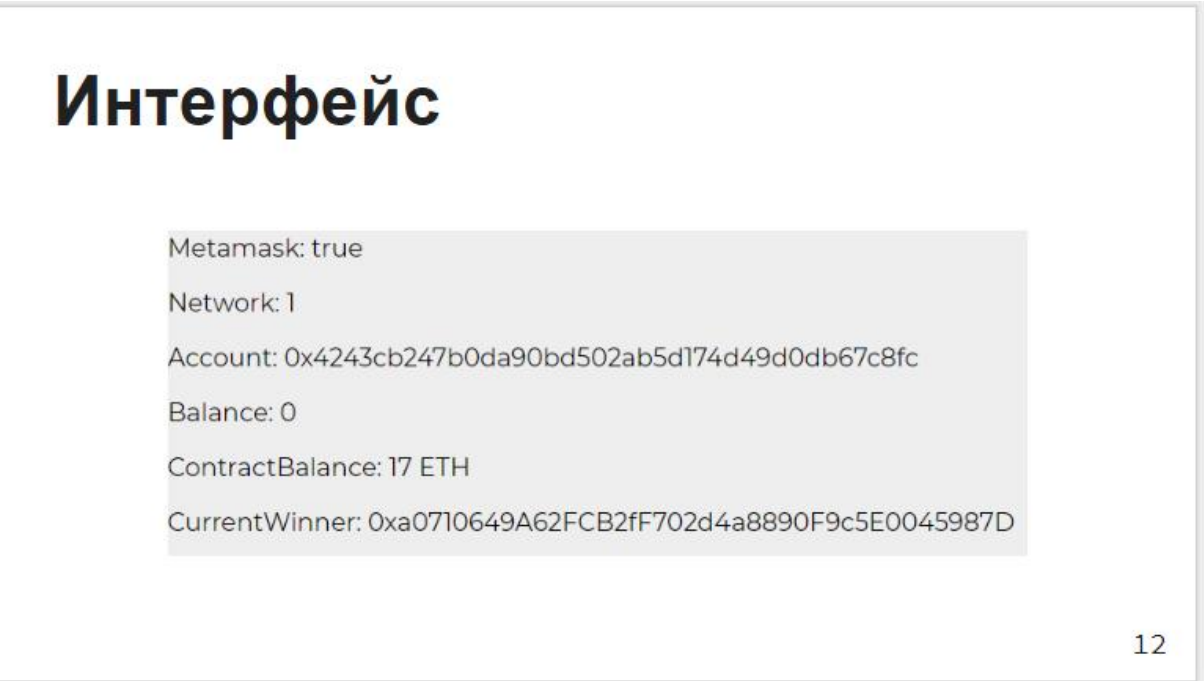
12

Рисунок А.12 - Плакат презентации 12



13

Рисунок А.13 - Плакат презентации 13



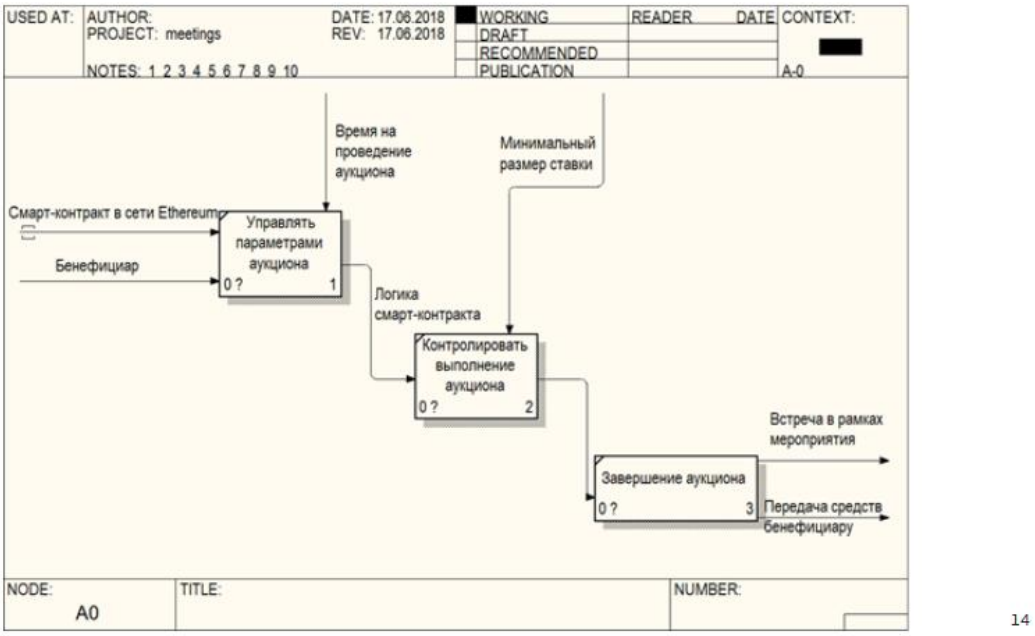


Рисунок А.12 - Плакат презентации 12

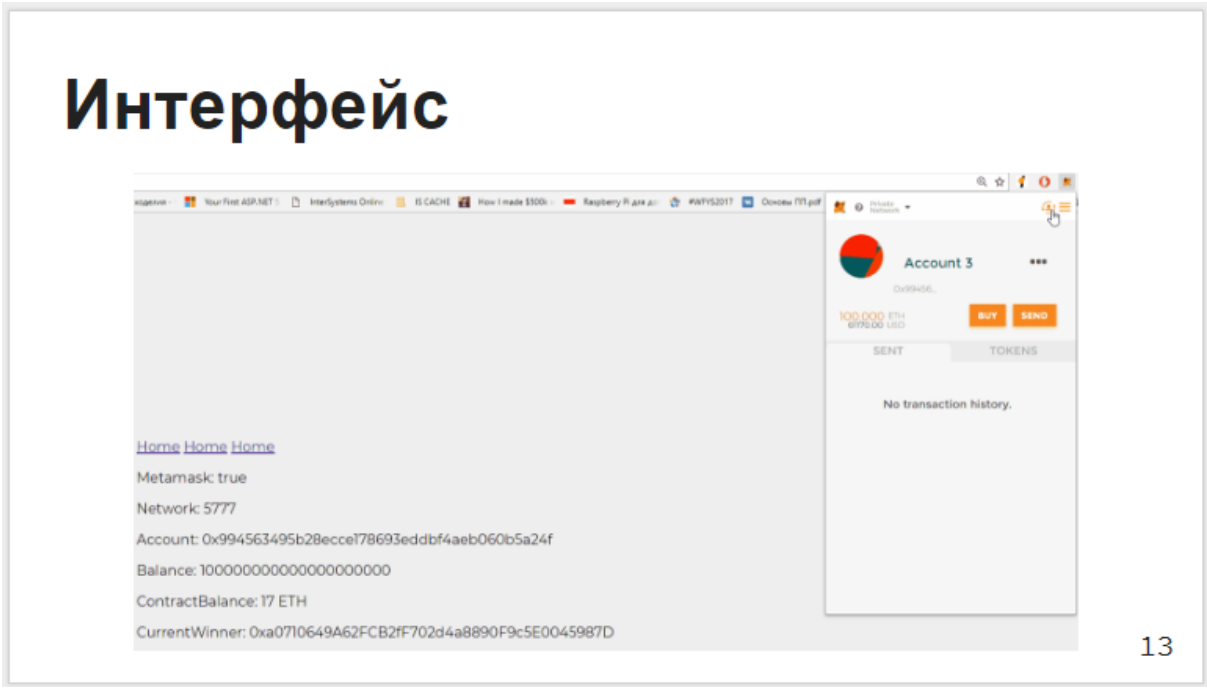


Рисунок А.13 - Плакат презентации 13

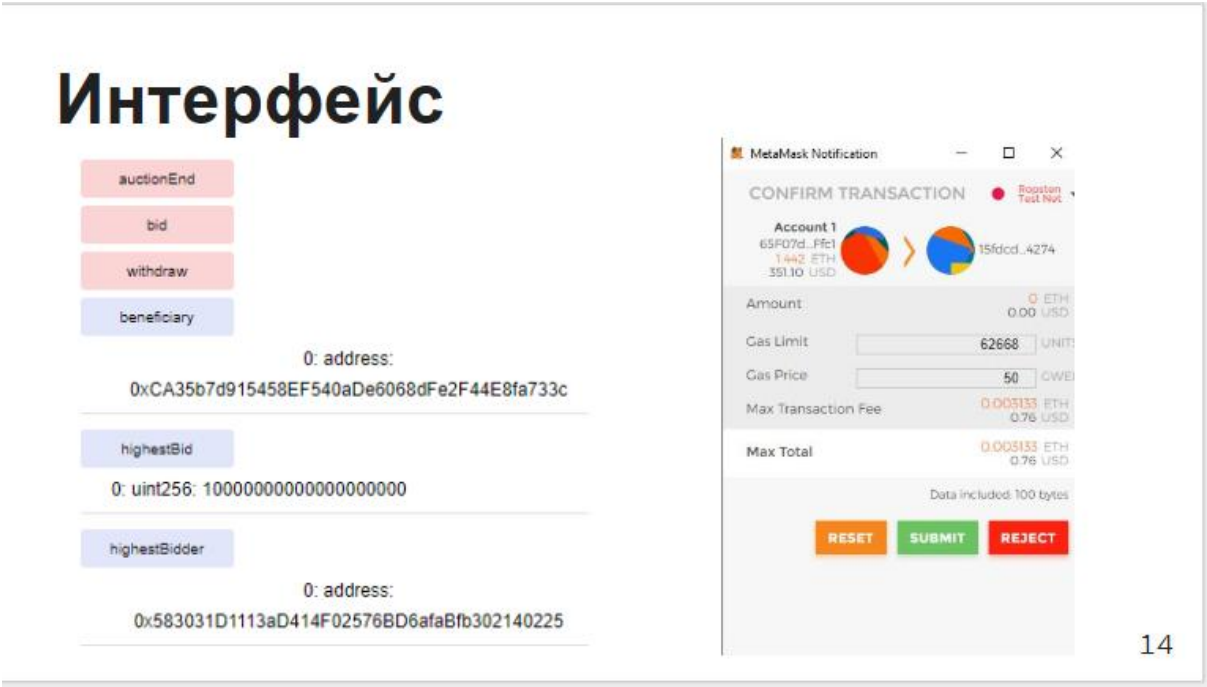


Рисунок А.14 - Плакат презентации 14

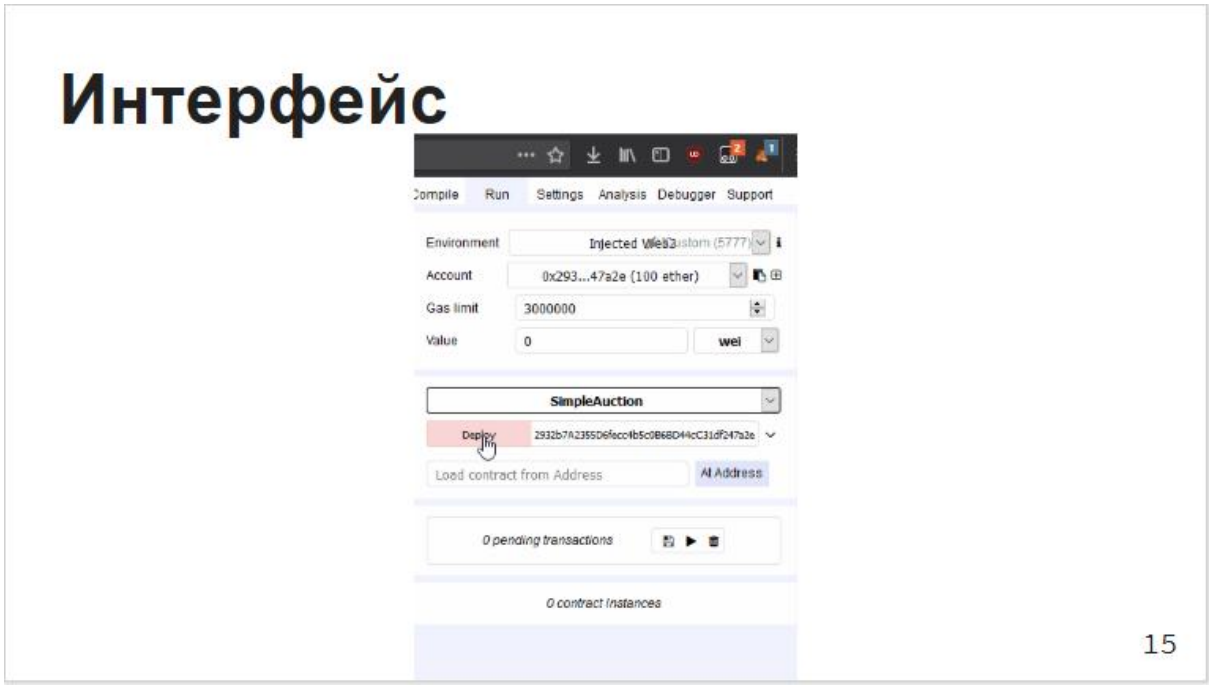


Рисунок А.15 - Плакат презентации 15

ПРИЛОЖЕНИЕ Б

Акт о внедрении